Massachusetts Institute of Technology
Instrumentation Laboratory
Cambridge, Massachusetts

LUMINARY Memo #120

To: Distribution
From: R. Larson
Date: 5 November 1969
Subject: C13STALL

Attached please find a memo from Bob Covelli describing in detail the program fix to the split radar pulse problem.

To: Russ Larson
From: Robert Covelli
Date: October 15, 1969
Subject: C13STALL - A Software Fix to the LGC - Radar Interface Problem

Summary

During the checkout of the lunar module prior to Apollo 10, a problem was found in the radar interface with the LGC. The problem is caused by the LGC writing into output channel 13 during the portion of a radar read sequence when the data is being shifted into the radar read counter. When this happens, it is possible for the data to be shifted one extra place, giving double weight to some of the bits causing an incorrect radar reading. Since the problem can only occur during the five millisecond interval when the radar sync pulses are being sent to the radar, the problem can be avoided by not writing into channel 13 during this period. Each time that the LGC initiates a radar read, a routine called RADSTART stores timing data which is used to determine the interval when the radar sync pulses are being transmitted. Whenever channel 13 is written into by a program that could possibly interfere with a radar reading, the routine C13STALL is called. If a radar reading is not being made, which is the normal situation, C13STALL returns to the calling program which than safely writes into channel 13. However, if a radar reading is being made, C13STALL uses the timing data stored by RADSTART to determine whether the radar sync pulses are being transmitted. If the pulses are not being transmitted, C13STALL returns to the calling program immediately; however, if the pulses are being sent, C13STALL "stalls" by waiting until the transmission of the radar sync pulses is completed. C13STALL then returns to the calling program which then safely writes into channel 13.

Radar Timing

The LGC requests a radar reading by setting bits 1-3 of channel 13 to indicate which radar function is desired, and setting bit 4 of channel 13 to 1. After bit 4 is set, the next F10A pulse generates a 100pps signal ADVCNT, which enables the generation of radar data request signals. Signal ADVCNT also advances a counter which generates signal CNTOF9 at the ninth ADVCNT signal. CNTOF9 inhibits the gereration of the radar request signals and enables the reception of the data. The next GTSET signal following

CNTOF9, five millisecond later causes the generation of RRSYNC or LRSYNC pulses at a 3200pps rate. The radar sync pulses shift the radar data into the radar input counter. The signal GTRST, which occurs five milliseconds after the radar sync pulses begin, generates the signal RADRPT, which inhibits the radar sync pulses and requests RUPT9. RUPT9 processes the radar data and completes the radar read.

Split Pulses

During the five milliseconds interval between the signals GTSET and GTRST following the signal CNTOF9, 15 radar sync pulses are sent to the radar. Each pulse is 3 microseconds wide and causes the radar to serially shift one bit of data into the radar input counter. In order to determine whether to send LRSYNC or RRSYNC pulses, the contents of bits 1 - 3 of channel 13 are used to gate the proper pulse. In order to send the LRSYNC pulse, bit 3 of channel 13 must be 1. In order to send the RRSYNC pulse, bit 3 must be 0 and bit 1 or bit 2 must be 1.

When a radar read is being made, the radar select bits (bits 1 - 3 of channel 13) remain in the same state as they were at the beginning of the reading, except for a very brief interval when the LGC writes into channel 13. When the LGC writes into any output channel, the entire channel is cleared for a 250 nanosecond period prior to writing the data into it. If this happens in channel 13 at the same time a radar sync pulse is being sent to the radar, the momentary clearing of bits 1 -3 remove the sync pulse for 250 nanoseconds, causing the pulse to be split into two pulses. The radar, responding to both pulses, shifts two bits of data. The LGC, however, cannot accept two pulses unless they are separated by at least six microseconds. As a result, the LGC accepts the logical OR of the two bits as one, and all subsequent bits shifted into the radar read counter are advanced by one, causing their weighting to be doubled.

Preventing Split Pulses

The problem in the Apollo 10 checkout occurred during the radar self test routine, which reads the radar once per second. The downlink interrupt,

DOWNRUPT, occurred once every 20 milliseconds, and wrote into channel 13 every time. The phasing was such that channel 13 was being written into when the radar sync pulses were being transmitted, causing periodic incorrect readings.

In order to reduce the probability of a split pulse in the Apollo 11 program, the downlink program was modified to that it would write into channel 13 only twice per second, rather than every 20 milliseconds. Although this reduced the probability of a split pulse, it did not assure satisfactory performance of the radar during the lunar landing programs. The problem was solved for Apollo 11 by attaching a blocking oscillator through ground support equipment connectors. The oscillator prevented the radar from recognizing a split pulse as two pulses.

The Apollo 12 LGC program, LUMINARY Revision 116, has a software solution to the problem. The LGC does not write into channel 13 during the transmission of radar sync pulses to the radar. The interval in which theses pulses are sent begins 85 milliseconds after the first F10A pulse following the setting of bit 4 of channel 13. The F10A pulse occurs every 10 milliseconds when the sixth bit of channel 4 is incremented. (Channel 4 contains the low and order output from the scaler, its least significant bit representing 1/3200 of a second.) Therefore the radar reading will begin DT channel 4 pulses after setting the radar activity bit, where DT is given by

DT = low 5 bits of (binary 100000 - low 5 bits of present contents of channel 4.)

Everytime the LGC requests a radar read, the routine RADSTART stores the negative of the channel 4 time in the register RADTIME, and stores the DT above in the register RADDEL. The routine C13STALL compares the present contents of channel 4 to RADTIME to determine wheter it is "safe" to write into channel 13. The comparison is made in the following way:

DeltaT = (present contents of channel 4 + RADTIME)

(Note - RADTIME is negative; also, the sum is overflow corrected in the event channel 4 overflowed since RADTIME was stored)

Then, if

Delta T $\geq$ 90 millisceonds + RADDELL

or Delta T $<$ 84.0625 milliseconds + RADDEL,

it is assumed safe to write into channel 13. Otherwise, Delta T is recomputed and the test is made again. After a maximum delay of 5.9325 milliseconds, Delta T will be greater than 90 ms + RADDEL, and the program which called C13STALL is permitted to write into channel 13. Using 84.0625 milliseconds as a lower limit is extremely conservative, allowing at least 625 microseconds for the return from C13STALL, writing into channel 13, and any counter interrupts.

Usage of RADSTART and C13STALL

Whenever the LGC initiates a radar read request, the follwing sequence must be performed:

```
INHINT
CA              (radar bits)
TC              IBNKCALL
CADR            RADSTART
```

RADSTART must be called by IBNKCALL even from its own bank because of its return.

Whenever a program that might interfere with a radar read writes into channel 13, the following sequence must be followed:

```
INHINT
TC              C13STALL
(perform write into channel 13)
RELINT
```

Some of the cases where C13STALL is not needed is when RADSTART initiates a radar read, or when a restart or fresh start initializes channel 13.

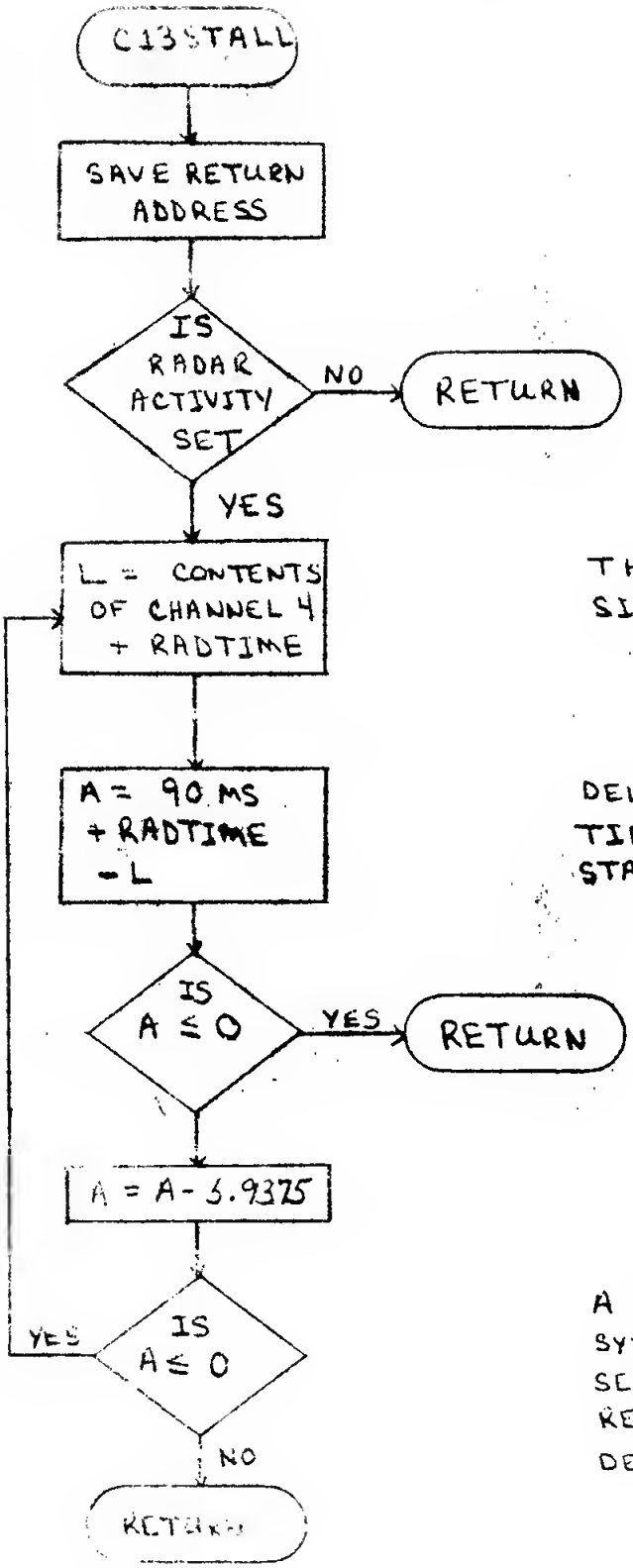
Testing the Software Fix

C13STALL has been shown to function properly by timing data taken from all digital simulations. It was also tested in the system test lab by running the radar self test with the downlink synchronized with the radar. There were no split pulses when C13STALL was used, while there were split pulses when it was not used in earlier revisions of the program. Clint Tillman of GAEC stated that there have been no split pulses observed during Grumman's testing of the program.

There have been no adverse effects observed due to the potential delay of programs writing into channel 13. Autopilot performance has not been degraded, and there has been no increase in the number of lost downrups.

Conclusion

The use of the routines RADSTART and C13STALL appears to be a satisfactory solution to the problem of bad radar readings due to split pulses. No problems have been detected in the use of this method.

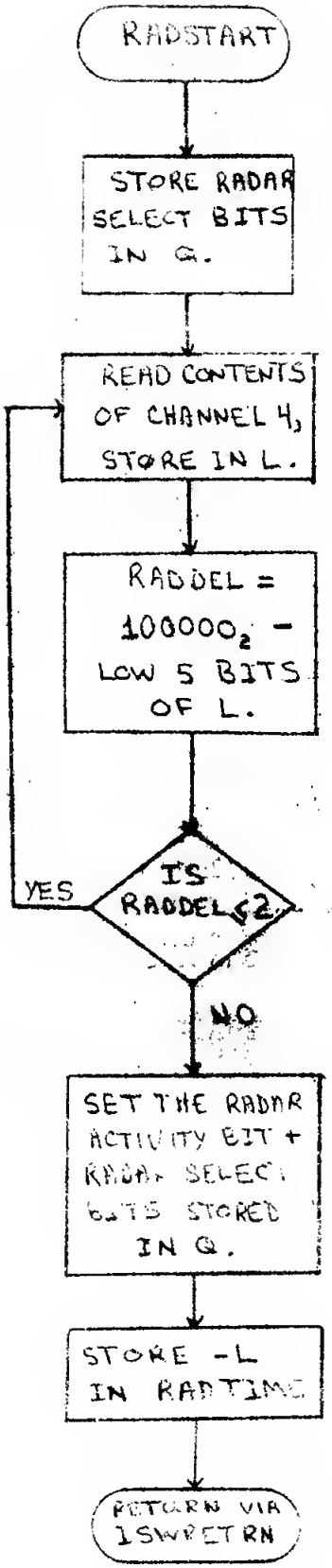


THIS IS THE DELTA TIME SINCE RADSTART BEGAN.

DELTA TIME - RADDEL IS THE TIME SINCE F10A PULSE STARTED RADAR READ SEQUENCE

RADAR SYNC PULSES HAVE BEEN SENT AND RADAR RUPT GENERATED

A ≤ 0 MEANS THAT RADAR SYNC PULSES ARE BEING SENT OR WILL BE VERY SOON. REPEAT ABOVE LOOP UNTIL DELTATIME - RADDEL ≥ 90 MS.



Inputs are the radar select bits in A and bits 1-4 of channel 13 zeroed.

RADDEL is the delta time until the next F10A pulse.

IF RADDEL ≤ 2, Then an F10A pulse might occur before the radar activity bit is set — therefore wait until the F10A pulse occurs. The maximum delay could be 937.5 microseconds

SAVE THE NEGATIVE TIME OF STARTING.

Distribution:

R. Ragan
D. Hoag
N. Sears
R. Battin
G. Cherry
A. Laats
G. Silver
R. Gilbert MIT/KSC
P. Felleman
M. Hamilton
B. McCoy
P. Adler
P. Rye
J. Kernan
D. Eyles
P. Volante
L. B. Johnson
J. Barker
T. Lawton MIT/MSC
G. Reasor MIT/MSC
K. Goodwin MIT/MSC
C. Tillman GAEC
T. Gibson FS5
T. Price FS5
C. Hackler EG2