



The Virtual AGC Project

Founded: Ron Burkey <info@sandroid.org>

When: 2003

Assisted since then by: Multitudes

Website: <http://www.ibiblio.org/apollo/>

Software: <https://github.com/virtualagc/virtualagc>

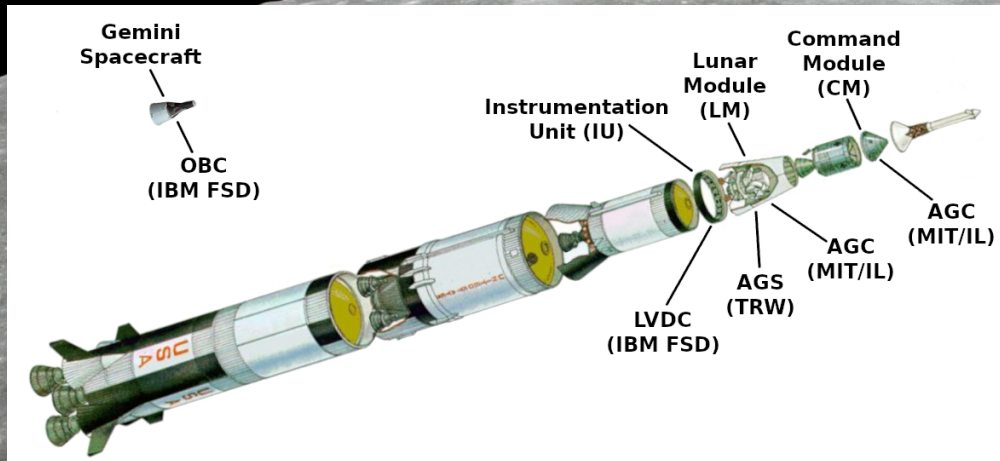
Scans: <https://archive.org/details/virtualagcproject>

^

I am Ron Burkey,
and I founded the Virtual AGC Project.
One evening in April of 2003,
while watching the movie Apollo 13,
I was struck by the idea of sharing
what I thought of as the "Apollo Experience".
To me, that meant running the original Apollo
flight-computer software on personal computers.
"Preserving" the software didn't occur to me,
because I IMAGINED finding it already online.



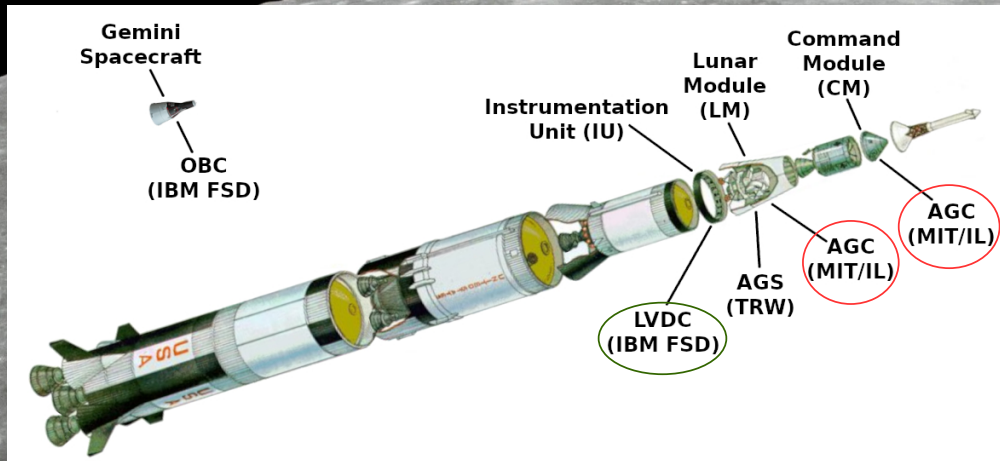
Computers, Spacecraft, and Saturn V



^
But I was wrong about that.
Nowadays, preserving software for the various
onboard computers of the Apollo and Gemini
Missions from the 1960's & 70's
is a big deal for me.



Computers, Spacecraft, and Saturn V

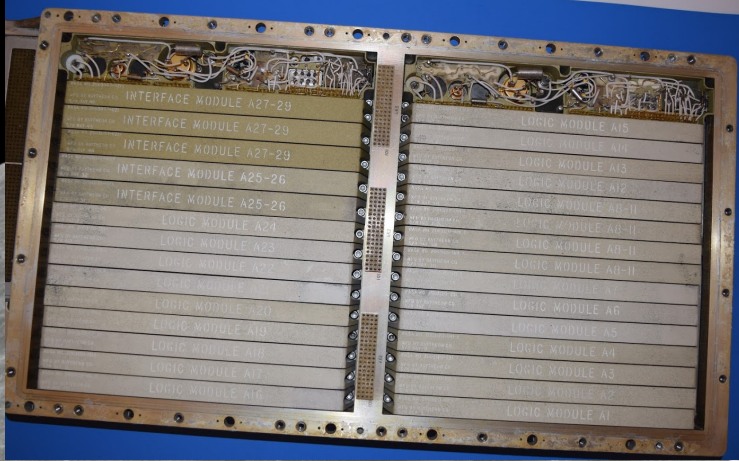


^

Today I'll concentrate on the Apollo Guidance Computer, or AGC. One was installed in the Apollo Command Module and another in the Lunar Module, or as it was originally called, the LEM. Later on I'll have a few remarks about the Saturn V's computer as well. The AGC was developed by MIT's Instrumentation Laboratory, today known as Draper Labs, and manufactured by the Raytheon Corporation. Because of time, I'll have to ignore the many interesting details of the AGC and its software, so I'll talk almost entirely about PRESERVATION of the software. But given that there were 20 Apollo missions, more or less, with separate software for the LEM vs the Command Module, that's still a LOT of software preservation to discuss.



AGC and DSKY



~ 62×32×15 cm, 32 kg



~ 22×20×18 cm, 8 kg

^
Here, on the left, we see the interior of an AGC. On the right, we see the astronauts' interface to the AGC, the Display Keyboard, or DSKY.

AGC Capabilities and Software



- 1's-complement arithmetic: ..., -1, -0, +0, +1, +2, ...
- 85 kHz clock speed
- ~ 78 kB read-only memory, in 36 banks
- ~ 4 kB read-write memory, in 8 banks
- No stack
- Ferrite-core based memory rather than semiconductor
- 15-bit word-size, plus 1 parity bit
- Intermixed assembly language & interpreter code
- Real-time, multi-tasking, fault-tolerant

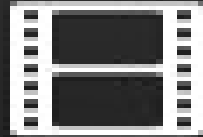


^

By modern standards, the AGC had a pretty novel design. It was very slow and had a very small memory. On the other hand, the real-time, multi-tasking, fault-tolerant executive software was quite sophisticated, given the limited hardware resources available.



Tidal Wave of AGC Software



^
In fact, nearly 2300 AGC software versions
have been identified!
We're looking at their evolutionary tree.
As far as I know, development began
with software called ECLIPSE in 1963.
Eventually, Command Modules were loaded with
software having names like CORONA, SUNSPOT,
SOLARIUM, SUNDISK, COLOSSUS,
COMANCHE, ARTEMIS, SKYLARK.
LEMs, meanwhile, had
SUNBURST, SUNDANCE, or LUMINARY.
And there were ground-test versions,
such as SUNDIAL and AURORA.
And engineering versions
like SHEPATIN, ZERLINA, and DIANA.



The Holy Grail: AGC-Related Project Goals

- Modern assembler ✓
- Modern CPU emulator ✓
- Spaceflight-simulator integration ✓
- Source code for *all* versions ...
- ... or at least all *flown* missions
- ... plus Apollo 1
- *All* relevant documentation



^

In my opinion, software preservation is about more than archiving source code.

It's about the documentation needed to understand the design, use, and evolution of the software.

It's about having access to software-development tools yourself.

Which in this case means being able to assemble the source code into an executable.

It's about being able to run the executable ... and to do so in the intended context.

Which in this case means to fly simulated but otherwise authentic Apollo missions.


How do you get the source material in the first place?

In a word - beg!

And you have to know who to beg, and how to beg.



Method 1: Starting From an Original Hardcopy



Source

Octals

GA11 ASSEMBLE REVISION 001 OF AGC PROGRAM LM99 BY NASA 2021112-061 16127 JULY 14,1969 PLY 1132 PAGE #18

LUNAR LANDING GUIDANCE EQUATIONS USER'S PAGE NO. 21 47 13

0850 REF 16 LAST 812 321373 03521 1 R

0851 REF 16 LAST 812 321374 77646 1 UNIT

0852 REF 16 LAST 812 321375 00017 1 STORE 140

0853 REF 16 LAST 812 321376 72441 0 OUT 541

0860 321377 00031 0 240

0861 REF 7 LAST 800 3213400 17474 0 STUOL HDGDISP

0862 3213401 00037 0 100

0863 3213402 41201 1 SL 00P

0864 3213403 20214 1 L10

0865 3213404 20474 0 HDGDISP

0866 3213405 45215 0 DAD DSU

0867 3213406 60065 0 360

0868 REF 5 LAST 805 3213407 02933 0 /LAND/

0869 REF 2 LAST 815 3213410 17775 1 STUOL HCALC1

0870 REF 9 LAST 818 3213411 03474 0 HDGDISP

0871 REF 2 LAST 122 3213412 86231 0 RDSU DIV

0872 REF 3 LAST 816 3213413 03845 0 VCGVERT

0873 REF 2 LAST 817 3213416 01237 0 TAUWHD

0874 3213415 51515 1 PVL ABVAL

0875 REF 7 LAST 817 3213416 01237 0 GDTFZ

0876 3213417 60471 0 DDV SPZ

0877 REF 2 LAST 809 3213420 09652 0 GSCALE

0878 3213421 00025 0 STORE 200

0879 3213422 77641 1 DAD

0880 3213423 41115 0 PEVL CALL

0881 REF 10 LAST 705 3213424 06620 0 UNITR

0882 REF 2 LAST 705 3213425 4764 1 CDUNMSH

0883 3213426 77641 1 DGT

0884 3213427 00017 1 JAU

0885 3213430 00027 1 STORE 220

0886 REF 6 LAST 817 3213431 45445 1 RDSU STAGE

0887 REF 4 LAST 809 3213432 50913 0 STUOL /APC/

0888 REF 6 LAST 817 3213435 03741 1 DELVWD

0889 3213434 53361 0 VASG VAD

0890 REF 2 LAST 817 3213435 24022 0 KAPF

0891 REF 3 LAST 817 3213436 02631 1 V6145

0892 3213437 63244 1 ABVAL HCOL

0893 REF 5 LAST 817 3213440 03762 1 THSTRIP

0894 3213441 45215 1 PDL

0895 REF 3 LAST 800 3213442 03760 0 DSU LASTTRIP

0896 REF 6 LAST 818 3213443 03762 1 THSTRIP

0897 REF 4 LAST 818 3213444 17760 0 STUOL LASTTRIP

0898 3213445 55271 0 DDV RDSU

0899 REF 1 3213446 55271 0 SHEPACT

0900 3213447 41325 0 PDL

0901 REF 7 LAST 800 3213450 03613 1 PHLIGHT

0902 REF 1 3213451 25544 1 DIV

0903 3213452 56273 0 HDG

0904 REF 7 LAST 809 3213453 01245 0 HSS

0905 REF 2 LAST 706 3213454 22066 1 SCLFFAC

UPDATE HDGDISP FOR RDSU 63.

GA11 ASSEMBLE REVISION 001 OF AGC PROGRAM LM99 BY NASA 2021112-061 16127 JULY 14,1969 PAGE 1710

ACTAL LISTING FOR PARAGRAPH # 210, WITH PARITY BIT IN BINARY AT THE RIGHT OF EACH WORD, *M* DENOTES UNUSED FIXED MEMORY

ALL VALID WORDS ARE BASIC INSTRUCTIONS EXCEPT THOSE MARKED *I* (INTERPRETER OPERATOR WORDS) OR *C* (CONSTANTS)

36:1000	C1 01 055 0	C1 37187 0	C1 00457 1	C1 03250 0	C1 77777 0	C1 77731 1	C1 00307 0	C1 11040 0
36:1010	C1 01 051 1	C1 05216 0	C1 77777 0	C1 77705 0	C1 00206 0	C1 30605 1	C1 00019 0	C1 14809 1
36:1020	C1 00 030 1	C1 00016 1	C1 03512 0	12325 0	12612 1	13954 1	15201 0	12146 1
36:1030	C1 00 000 1	C1 03770 1	C1 04087 1	12376 0	12963 1	12921 1	C1 01490 1	12329 0
36:1040	12184 1	13067 0	15261 0	12146 1	C1 06300 0	C1 03866 1	C1 74908 1	12240 1
36:1050	12563 1	12904 0	13123 0	12151 1	C1 77776 1	C1 03376 0	C1 74906 1	12400 0
36:1060	12567 0	C1 01450 1	12822 1	12611 1	13047 0	15261 0	C1 03120 1	12146 1
36:1070	C1 03 056 1	C1 74006 1	12166 1	12961 1	12961 1	13400 0	C1 01476 0	12329 0
36:1100	12162 1	12777 0	15261 0	12151 1	C1 04300 0	C1 03770 1	C1 66007 1	12376 0
36:1110	12963 1	12495 0	C1 01477 1	12395 0	12612 1	13062 1	15261 0	12120 1
36:1120	12121 0	12122 0	13123 1	12376 0	12963 1	12931 1	C1 04024 0	12376 0
36:1130	34755 1	55507 0	95510 0	04610 1	C1 7347 1	00006 1	31442 1	29512 1
36:1140	03004 0	04674 0	C1 79564 1	00003 1	51455 1	12009 0	44762 1	04616 1
36:1150	C1 76464 0	04037 0	I1 45345 1	C1 01644 0	C1 35145 1	C1 34041 0	C1 01124 0	I1 40014 0
36:1160	C1 03347 1	C1 74200 0	C1 27243 0	I1 64375 1	C1 00025 0	C1 01374 0	I1 77762 1	C1 25776 0
36:1170	C1 00017 1	I1 44932 0	C1 01234 0	C1 35720 1	C1 07130 1	C1 15317 0	C1 00015 0	C1 00041 1
36:1200	I1 71624 1	C1 27557 0	12211 0	00006 1	31561 1	53440 0	00006 1	31843 1
36:1210	21442 0	52155 1	93500 1	00008 1	43756 1	21500 1	00006 1	31500 0
36:1220	05277 0	C1 02240 0	C1 74087 0	05153 1	C1 02254 0	05221 1	C1 00077 1	15155 1
36:1230	44757 1	55163 0	06037 0	I1 51575 1	C1 03553 1	C1 03472 0	I1 77776 1	15155 1
36:1240	31756 0	05173 1	C1 02276 0	05353 1	C1 40154 0	44752 1	55163 0	51455 1
36:1250	42006 0	00006 1	63261 1	33146 1	05173 1	C1 03266 1	36027 1	05072 1
36:1260	C1 02263 1	C1 74067 0	15261 0	04610 1	C1 04610 1	13153 1	36027 1	05072 1
36:1270	C1 02273 0	C1 74067 0	15261 0	33761 1	04610 1	C1 20455 1	33444 0	05173 1
36:1300	C1 01 352 1	44762 1	55163 0	51455 1	30006 1	00006 1	62325 1	55477 0
36:1310	03173 1	C1 02300 1	36245 1	54001 1	46245 0	52753 1	40225 1	30018 1
36:1320	51455 1	10001 1	40106 1	7437 1	26106 1	00006 1	51455 1	30018 1
36:1330	50253 0	34755 0	54001 1	44752 1	52761 0	46025 1	55061 0	00006 1
36:1340	34755 1	92755 1	10763 1	15261 0	04635 0	C1 77410 1	02457 1	05353 1
36:1350	C1 00 051 0	15261 0	00006 1	34755 1	52757 0	33754 0	05173 1	C1 02403 1
36:1360	05316 0	C1 00153 0	05516 0	C1 00194 1	34940 1	10011 0	10757 0	12376 0
36:1370	34736 1	05105 0	C1 02540 1	C1 56067 0	05353 1	C1 00053 1	44760 1	55163 0

^

But at the moment, I'd prefer to talk about what happens AFTER we have the source materials in hand. Understand first that nobody EVER hands us machine-readable source code. We have to come up with that ourselves. Depending on the situation, so far we've used 3 different methods to do that. Let's start with the most-common scenario. It starts by getting access to a hard-copy of a so-called "assembly listing".



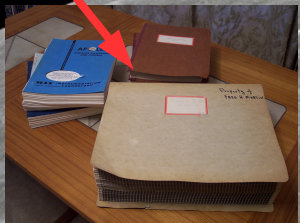
Method 1: Starting From an Original Hardcopy

Source

UPDATE HDGTDISP FOR K00H 63.

Octals

L		LUNAR LANDING GUIDANCE EQUATIONS		USER'S PAGE NO., 21		16127 JULY 14, 1969		PAGE #18	
0850	REF 16 LAST 812	32+3373	03521 1	R					
0851		32+3374	77656 1	UNIT					
0852		32+3376	00017 1	STORE 140					
0853		32+3376	72441 0	OUT 541					
0860		32+3377	00031 0	240					
0861	REF 7 LAST 800	32+3400	17474 0	STUDL HDGTDISP					
0862		32+3401	00037 0	100					
0863		32+3402	41261 1	SL 08P					
0864		32+3403	6014 1	L10					
0865		32+3404	02474 0	HDGTDISP					
0866		32+3405	45215 0	DAD 05U					
0867		32+3406	00065 0	360					
0868	REF 5 LAST 805	32+3407	02933 0	/LAND/					
0869	REF 2 LAST 815	32+3410	17775 1	STUDL HCALL3					
0870	REF 9 LAST 818	32+3411	03474 0	HDGTDISP					
0871	REF 2 LAST 822	32+3412	06231 0	ROBU DIV					
0872	REF 3 LAST 816	32+3413	03845 0	VDGVERT					
0873	REF 2 LAST 817	32+3416	01237 0	TBURGH					
0874		32+3415	51515 1	PVUL ABVAL					
0875	REF 7 LAST 817	32+3416	01237 0	GOTFZ					
0876	REF 2 LAST 809	32+3417	60471 0	DOV SP2					
0877	REF 2 LAST 809	32+3420	09532 0	-GSCALE					
0878		32+3421	06025 0	STORE 200					
0879		32+3422	77615 0	DAD					
0880		32+3423	45115 0	PEVL CALL					
0881	REF 10 LAST 705	32+3424	06520 0	UNIT 7E					
0882	REF 2 LAST 705	32+3425	47641 0	COU-NSH					
0883		32+3426	77641 1	OUT					
0884		32+3427	00017 1	140					
0885		32+3430	00027 1	STORE 220					
0886		32+3431	45445 0	ROBU STAGE					
0887	REF 4 LAST 809	32+3432	50913 0	STUDL /AFZ					
0888	REF 6 LAST 817	32+3433	03747 1	DELVARD					
0889		32+3434	53361 0	VASK VAD					
0890	REF 2 LAST 817	32+3435	24002 0	KAP1					
0891	REF 3 LAST 817	32+3436	02631 1	V81A5					
0892		32+3437	65244 1	ABVAL					
0893	REF 5 LAST 817	32+3440	03762 1	PSUL					
0894	REF 3 LAST 800	32+3442	03760 0	DSU					
0895	REF 6 LAST 818	32+3443	03742 1	LASTTRIP					
0896	REF 4 LAST 818	32+3444	17760 0	STUDL LASTTRIP					
0898		32+3445	55271 0	DOV ROBU					
0899	REF 1	32+3446	25515 0	SHPFACT					
0900		32+3447	41325 0	POUL DWP					
0901	REF 7 LAST 800	32+3450	03641 0	PNEIGHT					
0902	REF 1	32+3451	25534 1	H1TH					
0903		32+3452	56271 0	DOV					
0904	REF 7 LAST 809	32+3453	01245 0	H835					
0905	REF 2 LAST 706	32+3454	22066 1	SAL EFAC					



^
 That's a printout made by the Apollo developers when THEY assembled the source code 50 or 60 years ago. It's usually a stack of 11 inch by 14 inch fan-fold paper a couple of inches thick.



Method 1: Starting From an Original Hardcopy



GAP1 ASSEMBLE REVISION 001 OF AGC PROGRAM LMY99 BY NASA 2021112-061 16127 JULY 14,1969 FLY #132 PAGE #18

LUNAR LANDING GUIDANCE EQUATIONS

USER'S PAGE NO. 21 47 13

UPDATE HDGDISP FOR MOON 63.

0850	REF 16	LAST 812	32:3373	03521 1	R
0851			32:3374	77646 1	UNIT
0852			32:3376	00017 1	STORE 140
0853			32:3376	72441 0	OUT 541
0860			32:3377	00031 0	240
0861	REF 7	LAST 800	32:3400	17474 0	STUDL HDGDISP
0862			32:3401	00037 0	100
0863			32:3402	41201 1	SL 09P
0864			32:3403	20214 1	110
0865			32:3404	02474 0	HDGDISP
0866			32:3405	45215 0	DAD 05U
0867			32:3406	00065 0	360
0868	REF 5	LAST 805	32:3407	02933 0	/LAND/
0869	REF 2	LAST 815	32:3410	17775 1	STUDL HCALL3
0870	REF 9	LAST 818	32:3411	03474 0	HDGDISP
0871			32:3412	86231 0	RO5U DIV
0872	REF 3	LAST 816	32:3413	03845 0	VDGVERT
0873	REF 2	LAST 817	32:3416	01237 0	TABUG
0874			32:3415	51515 1	PVUL ABVAL
0875	REF 7	LAST 817	32:3416	01237 0	GOTFZ
0876			32:3417	60471 0	DDV SP2
0877	REF 2	LAST 809	32:3420	09650 0	-GSCALE
0878			32:3421	00025 0	STORE 200
0879			32:3422	77615 0	DAD
0880			32:3423	41115 0	PVUL CALL
0881	REF 10	LAST 792	32:3424	06520 0	UNIT 7E
0882	REF 2	LAST 705	32:3425	47641 0	CDU-MSH
0883			32:3426	77641 1	OUT
0884			32:3427	00017 1	140
0885			32:3430	00027 0	STORE 220
0886	REF 6	LAST 817	32:3433	03747 1	RO5U STAGE
0887	REF 4	LAST 809	32:3432	50913 0	STUDL /AFZ/
0888			32:3435	03747 1	DELVARD
0889			32:3434	53361 0	VASC VAD
0890	REF 2	LAST 817	32:3436	24020 0	KAP1
0891	REF 3	LAST 817	32:3436	02631 1	V6145
0892			32:3437	05244 0	ABVAL H5UL
0893	REF 1	LAST 817	32:3440	03762 1	THISTRIP
0894	REF 3	LAST 806	32:3442	03760 0	DSU
0895	REF 6	LAST 818	32:3443	03762 1	LASTTRIP
0896	REF 4	LAST 818	32:3444	17760 0	STUDL LASTTRIP
0897			32:3445	55271 0	DDV RUDY
0898	REF 1		32:3446	25515 0	SHEFACT
0899	REF 7	LAST 800	32:3447	41325 0	PDUL DHP
0900			32:3450	03611 0	PNEIGHT
0901	REF 1		32:3451	25514 1	HITIH
0902			32:3452	56271 0	DDV
0903			32:3453	01245 0	H555
0904	REF 7	LAST 809	32:3453	01245 0	SAL EFAC
0905	REF 2	LAST 796	32:3454	22066 1	

Source

GAP1 ASSEMBLE REVISION 001 OF AGC PROGRAM LMY99 BY NASA 2021112-061 16127 JULY 14,1969 FLY #132 PAGE #18

ACTUAL LISTING FOR PARAGRAPH # 210, WITH PARITY BIT IN BINARY AT THE RIGHT OF EACH WORD, *M* DENOTES UNUSED FIXED MEMORY

ALL VALID WORDS ARE BASIC INSTRUCTIONS EXCEPT THOSE MARKED *I* (INTERPRETIVE OPERATOR WORDS) OR *C* (CONSTANTS)

36:2000	C1 01 055 0	C1 37187 0	C1 09457 1	C1 03250 0	C1 77777 0	C1 77731 1	C1 00307 0	C1 11040 0
36:2010	C1 00151 1	C1 05216 0	C1 77777 0	C1 77705 0	C1 00206 0	C1 30605 1	C1 09019 0	C1 14809 0
36:2020	C1 00200 1	C1 00016 1	C1 03512 0	12325 0	12612 1	13952 1	15201 0	12146 1
36:2030	C1 00000 1	C1 03770 1	C1 04087 1	12376 0	12963 1	12921 1	C1 01490 1	12329 0
36:2100	12614 1	13067 0	15261 0	12146 1	C1 06300 0	C1 03866 1	C1 74908 1	12240 1
36:2105	12563 1	12904 0	13123 0	12151 1	C1 77776 1	C1 03376 0	C1 74906 1	12400 0
36:2106	12567 0	C1 01450 1	12822 1	12614 1	13047 0	15261 0	12246 1	C1 09120 0
36:2170	C1 03566 1	C1 74006 1	12166 1	12961 1	13400 0	C1 01476 0	C1 01476 0	12329 0
36:2140	12612 1	12777 0	15261 0	12151 1	C1 04300 0	C1 03770 1	C1 66007 1	12376 0
36:2110	12963 1	12455 0	C1 01477 1	12395 0	12612 1	13062 1	15261 0	12120 1
36:2120	12121 0	12122 0	13123 1	12376 0	12963 1	12921 1	C1 04024 0	C1 04024 0
36:2130	34755 1	55507 0	95510 0	04610 1	C1 73747 1	00006 1	31442 1	29512 1
36:2140	05004 0	04674 0	C1 79564 1	00003 1	51455 1	12009 0	44762 1	04616 1
36:2150	C1 76464 0	04037 0	I1 45345 1	C1 01640 1	C1 35145 1	C1 34041 0	C1 01124 0	I1 45014 0
36:2160	C1 03347 1	C1 74200 0	C1 27243 0	I1 64375 1	C1 00025 0	C1 01374 0	I1 77762 1	C1 01374 0
36:2170	C1 00017 1	I1 44332 0	C1 01254 0	C1 35701 1	C1 07130 1	C1 15317 0	C1 00015 0	C1 00011 1
36:2200	I1 27624 1	C1 27557 0	12211 0	00006 1	31561 1	53442 0	00006 1	31843 1
36:2210	21442 0	52155 1	93500 1	00008 1	43756 1	21500 1	00006 1	31500 0
36:2220	05277 0	C1 02240 0	C1 74087 0	05153 1	C1 02254 0	05521 1	C1 00077 1	15155 1
36:2230	44757 1	55163 0	06037 0	I1 51575 1	C1 03553 1	C1 03472 0	I1 77776 1	15155 1
36:2240	31756 0	05173 1	C1 02276 0	05353 1	C1 40154 0	44752 1	55153 0	51455 1
36:2250	42006 0	00006 1	63261 1	33145 1	05173 1	C1 03266 1	35027 1	05072 1
36:2260	C1 02263 1	C1 74067 0	15261 0	04610 1	C1 04056 1	13151 1	35027 1	05072 1
36:2270	C1 02273 0	C1 74067 0	15261 0	33761 1	04615 1	C1 20455 1	35344 0	05173 1
36:2300	C1 02352 1	44762 1	55163 0	51455 1	30006 1	00006 1	62325 1	55477 0
36:2310	03173 1	C1 02190 1	36245 1	54001 1	46245 0	42753 1	40225 1	35051 1
36:2320	51455 1	10001 1	40106 1	74237 1	26106 1	00006 1	51455 1	30010 1
36:2330	50253 0	34755 0	54001 1	44752 1	52761 0	46025 1	55061 0	00006 1
36:2340	34755 1	92755 1	10763 1	15261 0	04635 0	C1 77410 1	02457 1	05353 1
36:2350	C1 00051 0	15261 0	00006 1	34755 1	92757 0	C1 33754 0	05173 1	C1 02403 0
36:2360	05316 0	C1 00153 0	C1 95510 0	C1 00194 1	51490 1	10011 0	10757 0	12376 0
36:2370	34736 1	05105 0	C1 02540 1	C1 56067 0	05353 1	C1 00053 1	44760 1	55163 0

Octals

^
About 1500 pages are devoted to the source code



Method 1: Starting From an Original Hardcopy



Source

GAP1 ASSEMBLE REVISION 001 OF AGC PROGRAM LMY99 BY NASA 2021112-061 16127 JULY 14 1969 FLY 1132 PAGE #18

LUNAR LANDING GUIDANCE EQUATIONS USER'S PAGE NO. 21 47 13

0850 REF 16 LAST 812 3213373 03521 1 R
0851 REF 16 LAST 812 3213374 77646 1 UNIT
0852 REF 16 LAST 812 3213375 00017 1 STORE 140
0853 REF 16 LAST 812 3213376 72441 0 OUT 541
0860 3213377 00031 0 240
0861 REF 7 LAST 800 3213400 17474 0 STUOL HODTDISP
0862 3213401 00037 0 100
0863 3213402 41201 1 SL GMP
0864 3213403 20214 1 L10 HODTDISP
0865 3213404 20474 0 0
0866 3213405 45215 0 DAD DSU
0867 3213406 60065 0 360
0868 REF 5 LAST 805 3213407 02933 0 /LAND/
0869 REF 2 LAST 815 3213410 17775 1 STUOL HCALC1
0870 REF 9 LAST 818 3213411 03474 0 HODTDISP
0871 3213412 56231 0 RDSU DIV
0872 REF 3 LAST 816 3213413 03845 0 VDGVERT
0873 REF 2 LAST 817 3213414 05541 0 TDRUG
0874 3213415 51515 1 PUVL ABVAL
0875 REF 7 LAST 817 3213416 01237 0 GDTFZ
0876 3213417 60471 0 DGV SPZ
0877 REF 2 LAST 809 3213420 09552 0 GSCALE
0878 3213421 06025 0 STORE 200
0879 3213422 77641 0 DAD
0880 3213423 45115 0 PEVL CALL
0881 REF 10 LAST 705 3213424 06520 0 UNITR
0882 REF 2 LAST 705 3213425 47641 0 CDUNMSH
0883 3213426 77641 1 DGT 140
0884 3213427 00017 1 STORE 220
0885 3213430 00027 1 RDSU STAGE
0886 REF 6 LAST 809 3213432 50913 0 STUOL /APC/
0887 REF 4 LAST 809 3213432 50913 0 BELVARD
0888 REF 6 LAST 817 3213434 53361 0 VASC VAD
0889 3213435 24022 0 KSPH
0891 REF 3 LAST 817 3213436 02631 1 V6155
0892 3213437 05244 1 HSQL
0893 REF 5 LAST 817 3213440 03762 1 THSTRIP
0894 3213441 05229 1 PDL
0895 REF 3 LAST 800 3213442 03760 0 LASTTRIP
0896 REF 6 LAST 818 3213443 03762 1 THSTRIP
0897 REF 4 LAST 818 3213444 17760 0 STUOL LASTTRIP
0898 3213445 55271 0 DGV RDSU
0899 REF 1 3213446 25515 0 SHIFACT
0900 3213447 41325 0 PDL DMP
0901 REF 7 LAST 800 3213450 09611 0 PHLIGHT
0902 REF 1 3213451 25514 1 H1TH
0903 3213452 56271 0 DGV
0904 REF 7 LAST 809 3213453 01245 0 HSS
0905 REF 2 LAST 706 3213454 22061 1 SCLFFAC

UPDATE HODTDISP FOR RDSU 63.

GAP1 ASSEMBLE REVISION 001 OF AGC PROGRAM LMY99 BY NASA 2021112-061 16127 JULY 14 1969 FLY 1132 PAGE #19

ACTUAL LISTING FOR PARAGRAPH # 210, WITH PARITY BIT IN BINARY AT THE RIGHT OF EACH WORD, DENOTES UNUSED FIXED MEMORY

ALL VALID WORDS ARE BASIC INSTRUCTIONS EXCEPT THOSE MARKED *I* (INTERPRETIVE OPERATOR *O*DS) OR *C* (CONSTANTS)

36:2000	C1 01056 0	C1 37187 0	C1 09457 1	C1 03250 0	C1 77777 0	C1 7771 1	C1 00307 0	C1 11040 0
36:2010	C1 00151 1	C1 05216 0	C1 77777 0	C1 77705 0	C1 00206 0	20625 1	C1 00019 0	C1 74703 1
36:2020	C1 00200 1	C1 00016 1	C1 03512 0	12325 0	12612 1	19562 1	15201 0	12146 1
36:2030	C1 00000 1	C1 03770 1	C1 04087 1	12376 0	12563 0	12521 1	C1 01490 1	12329 0
36:2040	12634 1	13067 0	15261 0	12146 1	C1 06300 0	C1 03866 1	C1 74908 1	12340 1
36:2050	12563 1	12504 0	12151 1	C1 7777 0	C1 03376 0	C1 74906 1	12400 0	
36:2060	12567 0	C1 01830 1	12522 1	12611 1	1300 0	15261 0	12146 1	C1 09120 0
36:2070	C1 03566 1	C1 74906 1	12166 1	12561 1	1261 1	13400 0	C1 01476 0	12329 0
36:2100	12612 1	12777 0	15261 0	12151 1	C1 04300 0	C1 03770 1	C1 66067 1	12376 0
36:2110	12563 1	12455 0	C1 01477 1	12376 0	12612 1	13062 1	15261 0	12120 1
36:2120	12121 0	12122 0	12123 1	12376 0	12503 1	12501 1	C1 04024 0	
36:2130	34755 1	55507 0	55510 0	04610 1	C1 7347 1	00006 1	31442 1	25512 1
36:2140	05004 0	04674 0	C1 75564 1	00003 1	51455 1	12009 0	44762 1	04616 1
36:2150	C1 76464 0	04037 0	I1 45345 1	C1 01642 0	C1 53145 1	C1 34041 0	C1 01124 0	I1 40014 0
36:2160	C1 03347 1	C1 74200 0	C1 27243 0	I1 64375 1	C1 00025 0	C1 01374 0	I1 77762 1	C1 25776 0
36:2170	C1 00017 1	I1 44332 0	C1 01254 0	C1 35720 1	C1 07130 1	C1 15317 0	C1 00015 0	C1 00041 1
36:2200	I1 27624 1	C1 27557 0	12211 0	00006 1	31561 1	53440 0	00006 1	31813 0
36:2210	21442 0	52155 1	93500 1	00008 1	43756 1	21500 1	00006 1	31500 0
36:2220	05277 0	C1 02240 0	C1 74087 0	05153 1	C1 02254 0	05221 1	C1 00077 1	15155 0
36:2230	44767 1	55163 0	06037 0	I1 51575 1	C1 03553 1	C1 03472 0	I1 77776 1	15155 1
36:2240	31756 0	05173 1	C1 02276 0	05353 1	C1 40154 0	44752 1	55153 0	51455 1
36:2250	42006 0	00006 1	63261 1	33145 1	05173 1	C1 03266 1	35027 1	05072 1
36:2260	C1 02263 1	C1 74067 0	15261 0	04610 1	C1 04056 1	13151 1	35027 1	05072 1
36:2270	C1 02273 0	C1 74067 0	15261 0	33761 1	33761 1	C1 20455 1	33444 0	05173 0
36:2300	C1 02352 1	44762 1	55163 0	51455 1	30006 1	00006 1	62325 1	55477 0
36:2310	03173 1	C1 02300 1	36245 1	54001 1	46245 0	52753 1	40225 1	30051 0
36:2320	51455 1	10001 1	40106 1	7437 1	26106 1	00006 1	51455 1	30010 0
36:2330	50253 0	34755 0	54001 1	44752 1	52761 0	46025 1	55061 0	00006 1
36:2340	34755 1	92755 1	10763 1	15261 0	04635 0	C1 77410 1	02457 1	05353 1
36:2350	C1 00051 0	15261 0	00006 1	34755 1	92757 0	33756 0	05173 1	C1 02403 0
36:2360	05160 0	C1 00153 0	C1 55510 0	C1 00136 1	51490 1	10011 0	10757 0	12376 0
36:2370	34736 1	05105 0	C1 02540 1	C1 56067 0	05353 1	C1 00053 1	44760 0	55163 0

Octals

^
and about 150 pages are a so-called "octal listing" of the executable produced by the assembler.



Method 1 Continued ... Digitization



or



^
How does the hard-copy turn into a scan?
You CAN use a commercial service ...
if you can afford it ...
and if you can trust the service to handle
an irreplaceable historical printout.



Method 1 Continued ... Digitization



or



^
Or ... you can use a digital camera.
I like hanging the printout vertically,
and triggering the camera by remote control.
But know how to properly configure
the camera's white balance settings!
Which unfortunately, I didn't.



Method 1 Continued ... Digitization



or



^
Nowadays, a book scanner may be better.
But there's a trick involved ...



Method 1 Continued ... Digitization



or



^
because the best path for fan-fold paper is
UNDER the scanner, which normally
would be sitting directly on the table.



Method 1 Continued ... Digitization



or



^
I use a homemade platform that leaves
a gap between the scanner and the table.



Method 1 Continued ... Digitization



or

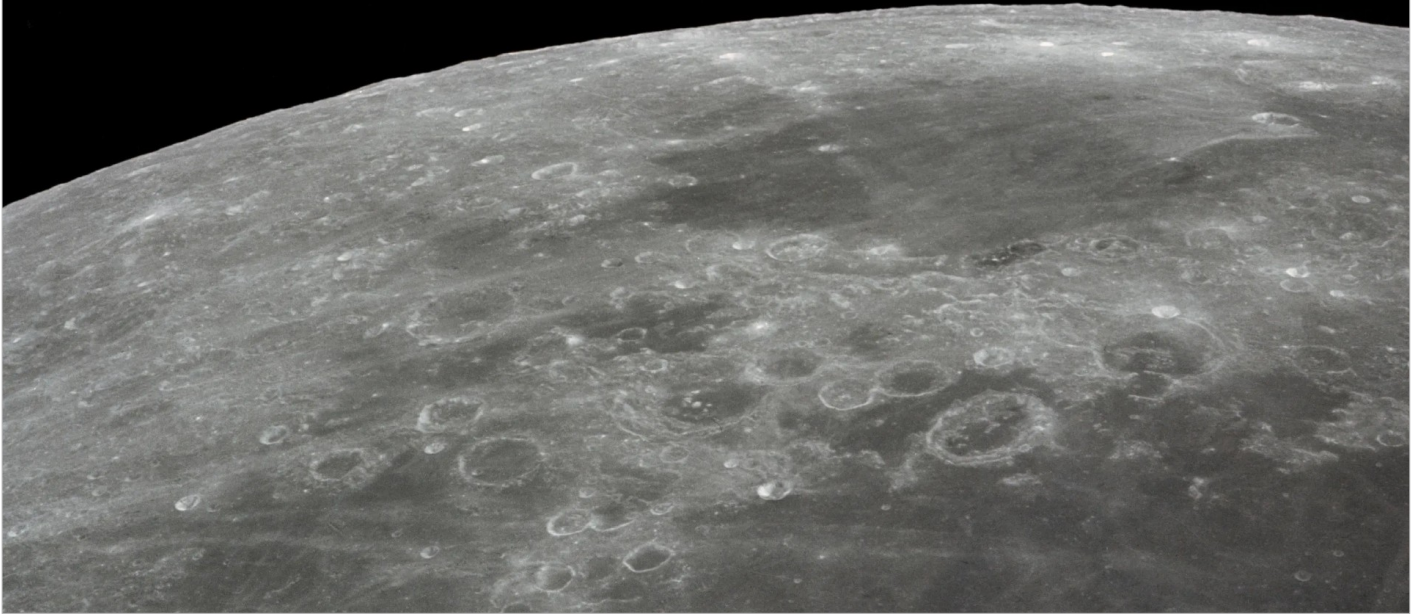


^
Also, a frame or registration marks
may be useful for software that post-processes
the images after you've scanned them.
Although ... these days, with the right setup,
your smartphone's camera may be even better
than a book scanner in some ways.



Method 1 Continued ...

Optical Character Recognition (OCR)?



^
Do we use optical-character recognition
software to turn the scans into
machine-readable source-code files?



Method 1 Continued ...

Optical Character Recognition (OCR)?



No CR!

^
No, we do not!
My slogan ... "OCR is no CR".
Why?



Method 1 Continued ...

Optical Character Recognition (OCR)?

Dreadful 60's-era printers

50 years of fading text, rips, folds, and stains

Text-obscuring horizontal green bars or black lines

OCR software's dictionaries are of no use

^
Well, OCR works best when you start with hard-copy having very good print quality. The printouts we work with seldom have that. Perhaps the worst thing is that modern OCR software seems to rely heavily on dictionaries specific to the languages being recognized. It's like auto-correct on your phone. And you know how well that works. Imagine how well it would work if your auto-correction dictionary was for the wrong language!



Method 1 Continued ... Transcription

GAP: ASSEMBLE REVISION 055 OF AGC PROGRAM CONANCHE BY NASA 2021113-051 10428 APR. 1, 1969 TROUBLE 043 PAGE 818
L P61-P67 USER'S PAGE NO. 30 E6 53

```
## Page 818
COUNT* 55/562.3
562.3  SETPD SLOAD
      0 ALFAPAD # ALFATRIM / 180, ALFA IS NEG.
      SRI PUSH # XCH PDL, COS TO PDL0
      COS PDDL # SIN TO PDL2
      SIN PDDL # SIN TO PDL2
      ROLLC
      VXSC
      COS UYA/2 #
      PDDL SIN # PUSH VECTOR INTO PDL4, 9 REF COORDS
      ROLLC
      VXSC VAD #
      UZA/2 # VECTOR FROM PDL4, 9 REF COORDS
      VSL1 STORE YNB # = UYD REF COORDS
      VXV VSL1 #
      UZA/2 # SIN TRIM FROM PDL2 REF COORDS
      VXSC PDDL # XCH PDL0 FOR COS TRIM
      VXSC VAD # FROM PDL0 REF COORDS
      UZA/2 # X SC AXIS (.5 UNIT) REF COORDS
      VSL1 STORE XNB # X SC AXIS (.5 UNIT) REF COORDS
      VXV VSL1 #
      YNB # Z SC IN REF COOR. SCALED AT 2
      STOVL ZNB # Z SC IN REF COOR. SCALED AT 2
      REF5MAT XSH
      STOVL REF5MAT +6
      YSH
      STOVL REF5MAT +12D
      STORE ZSH
      CLEAR GOTO # CAUSE CALCGA TO STORE ANS IN TP CPMT
      CRIFLAG CALCGA # CALCGA WILL RETURN TO ORIGINAL CALLER
      # VIA OPRET WITH 2.5 COMP. ANGLES IN CPMT
```

COUNT* 55/562.3		: p. 1631, 10,2400, V		
10,2375	67201 0	562.3	SETPD SLOAD	03012 41542 65346 65356 03316 74346 03550 73525
10,2377	00001 0		0	03316 53361 03556 77772 02722 76435 03542 65361
2 LAST 793	10,2400	03012 1	ALFAPAD ALFATRIM /180 + ALFA IS NEG.	53361 03542 77772 02714 76435 02722 26730 01736
10,2401	41542 1		SRI PUSH	26672 01744 26700 01752 02706 52014 00260 47311
10,2402	65346 0		COS PDDL XCH PDL, COS TO PDL0	
10,2403	65356 1		SIN PDDL SIN TO PDL2	
8 LAST 793	10,2404	03316 0	COS PDDL ROLLC	00004 52071 34371 05134 03441 60102 52071 00003
10,2405	74346 0		VXSC UYA/2	52006 22073 30065 75023 60000 54061 30065 75072
2 LAST 118	10,2406	03550 1	PDDL SIN PUSH VECTOR INTO PDL4, 9 REF	00006 75004 54062 30065 74105 56065 75066 10090
10,2407	73525 1		ROLLC	15426 30062 50061 54751 10066 12511 12477 40072
2 LAST 118	10,2411	53361 0	VXSC VAD	
10,2412	03556 1		UYA/2 VECTOR FROM PDL4, 9 REF	
10,2413	77772 0		VSL1 STORE YNB = UYD REF	22072 50071 52751 10066 12505 12511 40025 50071
5 LAST 763	10,2414	02722 1	VXV VSL1	55051 10065 12525 12530 40025 50061 55051 30065
10,2415	76435 1		UYA/2	62523 10000 12523 77777 12530 00006 30025 53136
3 LAST 118	10,2416	03542 1	VXSC PDDL	40062 50061 54750 30002 22073 00003 52006 22073
10,2417	65361 0			
10,2420	53361 0		VXSC VAD	30062 50061 54751 30070 50061 55052 00006 30064
10,2421	03542 1		UYA/2 XCH PDL0 FOR COS TRIM REF	50061 53435 12474 05536 13643 54155 35001 12722
10,2422	77772 0		VSL1 STORE XNB X SC AXIS (.5 UNIT) REF	33727 12722 54155 33730 12722 54155 35001 12700
9 LAST 774	10,2423	02714 1	VXV VSL1	54155 33253 13061 54155 33742 13061 54155 33727
10,2424	76435 1		YNB	13061 35017 03157 30100 73743 10000 12654 30100
10,2425	01736 1		REF5MAT	75007 00006 12624 15217 00004 45022 70100 65017
33 LAST 304	10,2427	01736 1	STOVL ZNB	54100 00003 41070 75013 10000 12640 40370 54370
10,2428	01736 1		REF5MAT +6	
35 LAST 818	10,2433	01752 0	STOVL YSH	35017 13040 30165 54156 50164 33753 54162 50164
10,2434	02706 1		REF5MAT +12D	31067 54160 54003 00002 30100 73476 10000 13676
3 LAST 463	10,2436	02706 1	STORE ZSH	40100 75015 00004 26100 12756 30370 74160 13342
10,2435	52016 0		CLEAR GOTO	54155 35021 12722 54155 35012 12722 54155 35021
10,2436	00200 0		CRIFLAG	54160 35021 13063 35021 54155 33726 12722 54155
3 LAST 774	10,2437	47311 1	CALCGA	35002 12722 54155 35002 12700 54155 33765 12722
				54155 35014 54160 03144 00003 12770 54155 33745
				13061 54155 33745 12722 35021 54164 03616 12744
				30100 73764 10000 13017 30100 77655 00006 12752
				35021 13216 30100 73762 00006 12760 35017 13216
				03156 03427 05201 03444 30025 55147 35021 13040
				30167 77674 54163 37670 05210 30160 73744 10000

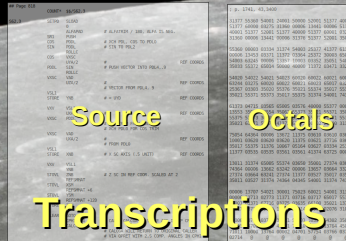
Transcribe source code

Transcribe octal listing

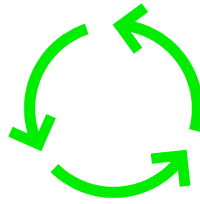
^
INSTEAD, a team of volunteers MANUALLY transcribes the source code, and SEPARATELY transcribes the octal listing.



Method 1 Continued ... Transcription



Assemble
the Source



Correct the
Transcription(s)

Compare the
Executables

^
We then process the transcribed source code using our modern assembler, and compare the executable produced by the assembler to the transcribed octal listing. If there are mismatches, we correct the transcription. And we just keep doing that until the executable is perfect.



Method 1 Continued ... Transcription

```

## Page B18
Source 55/562.3
562.3 SETPO SLOAD
      0 ALFAPAD
SR1 PDSH # ALFATRIM / 180, ALFA IS NEG.
COS PDDL # XCH POL., COS TO PDL0
SIN PDDL # SIN TO PDL2
      ROLLC
COS VVSC
PDDL UXA/2 # REF COORDS
SIN # PUSH VECTOR INTO PDL4, 9
VVSC VAD # REF COORDS
VAD UXA/2 # VECTOR FROM PDL4, 9
VSL1 STORE YNB # = UVD REF COORDS
VVV VSL1 # REF COORDS
UXA/2 # SIN TRIM FROM PDL2
VVSC PDDL # XCH PDL0 FOR COS TRIM
VAD VAD # REF COORDS
UXA/2 # FROM PDL0
VSL1 STORE XNB # X SC AXIS (.5 UNIT) REF COORDS
VVV VSL1 YNB
ZNB # Z SC IN REF COOR. SCALED AT 2
STOVL XSH REFSPMAT
REFSPMAT +6
YSH REFSPMAT +120
STORE ZSH
CLEAR
GOTO CRNFLAG
CRNFLAG CALCGA # CAUSE CALCGA TO STORE ANS IN TP CPNI
CALCGA # CALCGA WILL RETURN TO ORIGINAL CALLER
# VSA UPRET WITH 2.5 COMP. ANGLES IN CPNI
00006 13707 54021 30001 75023 60021 54001 31376
00006 13713 02773 11371 03716 03717 65017 55371
40001 61371 67716 03275 03635 66100 35021 13735
35017 13735 35016 13735 36214 54331 02076 31011
00006 13743 03753 31011 54301 30331 54302 26017
54303 04701 57573 40101 75010
71011 10000 13764 00002 04701 07754 02076 02077
02714

```

- 1) Transcribe source code
- 2) Transcribe octal listing
- 3) Proofread like crazy!
- 4) Assemble
- 5) Debug
- 6) Repeat until assembly matches octal listing

```

Page B18 Assembly listing generated by yaYUL
036474,001193: 10,2376
036475,001194:
036476,001195: 10,2376 67201
036477,001196: 10,2377 00001
036478,001197: 10,2400 03012
036479,001198: 10,2401 41542
036480,001199: 10,2402 05346
036481,002200: 10,2403 05356
036482,001201: 10,2404 05316
036483,002202: 10,2405 74346
036484,002203: 10,2406 03550
036485,002204: 10,2407 75225
036486,002205: 10,2410 03516
036487,002206: 10,2411 53361
036488,002207: 10,2412 02556
036489,002208:
036490,002209: 10,2413 77772
036491,002210: 10,2414 02722
036492,002211:
036493,002212: 10,2415 76435
036494,002213: 10,2416 05542
036495,002214: 10,2417 05561
036496,002215:
036497,002216:
036498,002217: 10,2420 53361
036499,002218: 10,2421 03542
036500,002219:
036501,002220: 10,2422 77772
036502,002221: 10,2423 02714
036503,002222:
036504,002223: 10,2424 76435
036505,002224: 10,2425 02722
036506,002225: 10,2426 26730
036507,002226: 10,2427 01296
036508,002227: 10,2430 26672
036509,002228: 10,2431 01144
036510,002229: 10,2432 26700
036511,002230: 10,2433 01752
036512,002231: 10,2434 02706
036513,002232:
036514,002233: 10,2435 52014
036515,002234: 10,2436 00260
036516,002235: 10,2437 47311
036517,002236:
036518,002237:
036519,002238:
036520,002239:
036521,002240:
COUNT* 55/562.3
SETPO SLOAD
      0 ALFAPAD
SR1 PDSH # ALFATRIM / 180, ALFA IS NEG.
COS PDDL # XCH POL., COS TO PDL0
SIN PDDL # SIN TO PDL2
      ROLLC
COS VVSC # REF COORDS
PDDL UXA/2 # VECTOR FROM PDL4, 9
SIN # PUSH VECTOR INTO PDL4, 9
VVSC VAD # REF COORDS
VAD UXA/2 # VECTOR FROM PDL4, 9
VSL1 STORE YNB # = UVD REF COORDS
VVV VSL1 # REF COORDS
UXA/2 # SIN TRIM FROM PDL2
VVSC PDDL # XCH PDL0 FOR COS TRIM
VAD VAD # REF COORDS
UXA/2 # FROM PDL0
VSL1 STORE XNB # X SC AXIS (.5 UNIT) REF COORDS
VVV VSL1 YNB
ZNB # Z SC IN REF COOR. SCALED AT 2
STOVL XSH REFSPMAT
REFSPMAT +6
YSH REFSPMAT +120
STORE ZSH
CLEAR
GOTO CRNFLAG
CRNFLAG CALCGA # CAUSE CALCGA TO STORE ANS IN TP CPNI
CALCGA # CALCGA WILL RETURN TO ORIGINAL CALLER
# VSA UPRET WITH 2.5 COMP. ANGLES IN CPNI

```

^
Here's a summary of the entire entire process.
Notice that syntax highlighting for source code
is a very convenient byproduct.



Method 1 Continued ... Transcription

```

Page 818
Source $S/562.3
562.3
SETPO SLOAD
0
ALFAPAD # ALFATRIM / 180, ALFA IS NEG.
SRI POSH # XCH POL., COS TO POLD
COS PDDL # SIN TO POL2
PDDL ROLLC
COS VVSC # REF COORDS
VVA/2 # PUSH VECTOR INTO POL4., 9
SIN #
PDDL # VECTOR FROM POL4., 9
VVA # = UVD # REF COORDS
VVA VV1 # REF COORDS
STORE YNB #
VXSC PDDL # SIN TRIM FROM POL2
VAD # XCH POLD FOR COS TRIM
UVA/2 # FROM POLD # REF COORDS
VXSC # X SC AXIS (.5 UNIT) # REF COORDS
VVA VV1 #
STORE YNB #
STOVL ZNB # Z SC IN REF COOR. SCALED AT 2
REFSPMAT XSH #
REFSPMAT +6 XSH #
REFSPMAT +120 YSH #
STORE ZSH #
CLEAR # CAUSE CALCGA TO STORE ANS IN TP CPNI
CPNIFLAG # CALCGA WILL RETURN TO ORIGINAL CALLER
CALCGA # VSA UPRET WITH 2.5 COMP. ANGLES IN CPNI
00006 13707 54021 30001 75023 60021 54001 31376
00006 13713 02773 11371 03716 03717 65017 55371
40001 61371 67716 03275 03635 66100 35021 13735
35017 13735 35016 13735 36214 54331 02076 31011
00006 13743 03753 31011 54301 30331 54302 26017
54303 04701 57573 40101 75010
71011 10000 13764 00002 04701 37754 03716 02777
02714 @ @ @ @ @ @ @ @ @ @

```

- 1) Transcribe source code
- 2) Transcribe octal listing
- 3) Proofread like crazy!
- 4) Assemble
- 5) Debug
- 6) Repeat until assembly matches octal listing

```

Page 818 Assembly listing generated by aYUL
036474,001193: 10,2376
036475,001194:
036476,001195: 10,2376
036477,001196: 10,2377
036478,001197: 10,2400
036479,001198: 10,2401
036480,001199: 10,2402
036481,002200: 10,2403
036482,001201: 10,2404
036483,002202: 10,2405
036484,002203: 10,2406
036485,002204: 10,2407
036486,002205: 10,2410
036487,002206: 10,2411
036488,002207: 10,2412
036489,002208:
036490,002209: 10,2413
036491,002210: 10,2414
036492,002211:
036493,002212: 10,2415
036494,002213: 10,2416
036495,002214: 10,2417
036496,002215:
036497,002216:
036498,002217: 10,2420
036499,002218: 10,2421
036500,002219:
036501,002220: 10,2422
036502,002221: 10,2423
036503,002222:
036504,002223: 10,2424
036505,002224: 10,2425
036506,002225: 10,2426
036507,002226: 10,2427
036508,002227: 10,2430
036509,002228: 10,2431
036510,002229: 10,2432
036511,002230: 10,2433
036512,002231: 10,2434
036513,002232:
036514,002233: 10,2435
036515,002234: 10,2436
036516,002235: 10,2437
036517,002236:
036518,002237:
036519,002238:
036520,002239:
036521,002240:
67201
00001
03012
41542
05346
05356
05316
74346
03550
73525
03516
53361
02556
77772
02722
76435
05542
05561
53361
03542
77772
02714
76435
02722
26730
01796
26672
01744
26709
01752
02706
52014
00260
47311
COUNT* $S/562.3
562.3
SETPO SLOAD
0
ALFAPAD # ALFATRIM / 180, ALFA IS NEG.
SRI POSH # XCH POL., COS TO POLD
COS PDDL # SIN TO POL2
PDDL ROLLC
COS VVSC # REF COORDS
VVA/2 # PUSH VECTOR INTO POL4., 9
SIN #
PDDL # VECTOR FROM POL4., 9
VVA # = UVD # REF COORDS
VVA VV1 # REF COORDS
STORE YNB #
VXSC PDDL # SIN TRIM FROM POL2
VAD # XCH POLD FOR COS TRIM
UVA/2 # FROM POLD # REF COORDS
VXSC # X SC AXIS (.5 UNIT) # REF COORDS
VVA VV1 #
STORE YNB #
STOVL ZNB # Z SC IN REF COOR. SCALED AT 2
REFSPMAT XSH #
REFSPMAT +6 XSH #
REFSPMAT +120 YSH #
STORE ZSH #
CLEAR # CAUSE CALCGA TO STORE ANS IN TP CPNI
CPNIFLAG # CALCGA WILL RETURN TO ORIGINAL CALLER
CALCGA # VSA UPRET WITH 2.5 COMP. ANGLES IN CPNI

```

^
This process insures that INSTRUCTIONS are transcribed correctly.



Method 1 Continued ... Transcription

```
## Page 818 55/562.3
562.3 SETPO SLOAD
0 ALFAPAD
SRI PDSH # ALFATRIM / 100, ALFA IS NEG.
COS PDDL # XCH POL., COS TO POL0
SIN PDDL # SIN TO POL2
ROLLC
COS VASC #
PDDL UXA/2 # PUSH VECTOR INTO POL4., 9 REF COORDS
SIN #
VASC # VECTOR FROM POL4., 9 REF COORDS
VAD UXA/2 #
VSL1 #
STORE YNB # = UVD REF COORDS
VVV VSL1 #
UXA/2 #
VASC PDDL # SIN TRIM FROM POL2
VASC VAD # XCH POL0 FOR COS TRIM
UXA/2 # FROM POL0 REF COORDS
VSL1 # FROM POL0
STORE XNB # X SC AXIS (.5 UNIT) REF COORDS
VSL1 #
VNB #
ZNB # Z SC IN REF COOR. SCALED AT 2
STOVL XSH REFSPMAT
REFSPMAT +6 3672
YSH REFSPMAT +6 3621
ZSH REFSPMAT +120 5373
STORE ZSH 9006
CLEAR
GOTO CRNFLAG
CRNFLAG 3664
CALCGA 3240
CALCGA # CALCGA WILL RETURN TO ORIGINAL CALLER
# VSA UPNET WITH 2.5 COMP. ANGLES IN CPNT 3545
# 4373

00006 13707 54021 30001 75023 60021 54001 31376
00006 13713 02773 11371 03716 03717 65017 55371
40001 61371 67716 03275 03635 66100 35021 13735
35017 13735 35016 13735 36214 54331 02076 31011

00006 13743 03753 31011 54301 30331 54302 35017
54303 04701 57573 40101 75010
71011 10000 13764 00002 04701 37754 03716 367
02774 e e e e e e e e
```

- 1) Transcribe source code
- 2) Transcribe octal listing
- 3) Proofread like crazy!
- 4) Assemble
- 5) Debug
- 6) Repeat until assembly matches octal listing

```
Page 818 Assembly listing generated by yaYUL
036474,001193: 10,2376 COUNT* 55/562.3
036475,001194: 67201 SLOAD
036476,001195: 10,2376 0
036477,001196: 10,2376 00001 ALFAPAD
036478,001197: 10,2400 03012
036479,001198: 10,2401 41542 SRI PUSH
036480,001199: 10,2402 05346 COS PDDL
036481,002200: 10,2403 05356 SIN PDDL
036482,001201: 10,2404 05316 ROLLC
036483,002202: 10,2405 74346 COS VASC
036484,002203: 10,2406 03550 UXA/2
036485,002204: 10,2407 73525 PDDL SIN
036486,002205: 10,2410 03516 ROLLC
036487,002206: 10,2411 53361 VASC VAD
036488,002207: 10,2412 02556 UXA/2
036489,002208: 77772 VSL1
036490,002209: 10,2413 02722 STORE YNB
036491,002210: 10,2414 02722
036492,002211: 76435
036493,002212: 10,2415 76435 VVV VSL1
036494,002213: 10,2416 05542 UXA/2
036495,002214: 10,2417 05561 VASC PDDL
036496,002215: 53361
036497,002216: 03542 VASC VAD
036498,002217: 10,2420 53361 UXA/2
036499,002218: 10,2421 03542
036500,002219: 77772 VSL1
036501,002220: 10,2422 02714 STORE XNB
036502,002221: 10,2423 02714
036503,002222: 02722 YNB
036504,002223: 10,2424 76435 VVV VSL1
036505,002224: 10,2425 02722 YNB
036506,002225: 10,2426 26730 STOVL ZNB
036507,002226: 10,2427 01296 REFSPMAT
036508,002227: 10,2430 26672 STOVL XSH
036509,002228: 10,2431 01144 REFSPMAT
036510,002229: 10,2432 26709 STOVL YSH
036511,002230: 10,2433 01152 REFSPMAT
036512,002231: 10,2434 02706 STOVL ZSH
036513,002232: 52014 CLEAR
036514,002233: 10,2435 52014 CRNFLAG
036515,002234: 10,2436 00260 CALCGA
036516,002235: 10,2437 47111
036517,002236:
036518,002237:
036519,002238:
036520,002239:
036521,002240:
```

```
# ALFATRIM / 100, ALFA IS NEG.
# XCH POL., COS TO POL0
# SIN TO POL2
#
# PUSH VECTOR INTO POL4., 9 REF COORDS
# VECTOR FROM POL4., 9 REF COORDS
# = UVD REF COORDS
#
# SIN TRIM FROM POL2
# XCH POL0 FOR COS TRIM
# FROM POL0 REF COORDS
# X SC AXIS (.5 UNIT) REF COORDS
#
# Z SC IN REF COOR. SCALED AT 2
#
# CAUSE CALCGA TO STORE ANS IN TP CPNT
# CALCGA WILL RETURN TO ORIGINAL CALLER
# VSA UPNET WITH 2.5 COMP. ANGLES IN CPNT
```

^
But what about program comments, which are discarded by the assembler, and thus NOT cross-checked by the assembly process? Typos in program comments may seem benign, but we'd still like some kind of extra proofing magic to eliminate them.



Method 1 Continued ...

Proofing Magic: Colorize!

- 1) Don't use OCR to extract text.
- 2) Instead use OCR to overlay the transcribed text in *color* atop the scan.
- 3) Anywhere there's color after that ... double check!

SFTEMP1 SFTEMP2 WITH THE DP SFINTAB ENTRIES..

25 S C O M M A N D S IN SFTEMP1.

```
NN NORMAL NOUNS
00 NOT IN USE
01 SPECIFY MACHINE ADDRESS (FRACTIONAL)
02 SPECIFY MACHINE ADDRESS (WHOLE)
03 SPECIFY MACHINE ADDRESS (DEGREES)
04 ANGULAR ERROR/DIFFERENCE
05 ANGULAR ERROR/DIFFERENCE
06 OPTION CODE
07 ECADR OF WORD TO BE MODIFIED
   1 TO SET OR 0 TO RESET SELECTED BITS
08 ALARM DATA
09 ALARM CODES
10 CHANNEL TO BE SPECIFIED
11 TIG OF CSI (HRS,MIN,SEC)
12 OPTION CODE
   (USED BY EXTENDED VERBS ONLY)
13 TIG OF CDH (HRS,MIN,SEC)
14 CHECKLIST
   (USED BY EXTENDED VERBS ONLY)
15 INCREMENT MACHINE ADDRESS
16 TIME OF EVENT (HRS,MIN,SEC)
17 SPARE
18 AUTO MANEUVER BALL ANGLES
19 SPARE
20 ICDU ANGLES
21 PIPAS
22 NEW ICDU ANGLES
23 SPARE
24 DELTA TIME FOR AGC CLOCK(HRS,MIN,SEC)
25 CHECKLIST
   (USED WITH PLEASE PERFORM ONLY)
26 PRIO/DELAY, ADRES, BBCON
27 SELF TEST ON/OFF SWITCH
```

^
I've already claimed that OCR isn't too useful for extracting source code from a scan. But it's not entirely worthless for proofing. By combining a scan, an OCR of the scan, and a transcription of the scan, there's a way to overlay a colorized version of the transcription directly atop the black text of the scanned image.



Method 1 Continued ...

Proofing Magic: Colorize!

- 1) Don't use OCR to extract text.
- 2) Instead use OCR to overlay the transcribed text in *color* atop the scan.
- 3) Anywhere there's color after that ... double check!

SFTEMP1 SFTEMP2 WITH THE DP SFINTAB ENTRIES..

25 S C U M M A T I E S I N S F T E M P 1 .

```
NN NORMAL NOUNS
00 NOT IN USE
01 SPECIFY MACHINE ADDRESS (FRACTIONAL)
02 SPECIFY MACHINE ADDRESS (WHOLE)
03 SPECIFY MACHINE ADDRESS (DEGREES)
04 ANGULAR ERROR/DIFFERENCE
05 ANGULAR ERROR/DIFFERENCE
06 OPTION CODE
07 ECADR OF WORD TO BE MODIFIED
   1 TO SET OR 0 TO RESET SELECTED BITS
08 ALARM DATA
09 ALARM CODES
10 CHANNEL TO BE SPECIFIED
11 TIG OF CSI (HRS,MIN,SEC)
12 OPTION CODE
   (USED BY EXTENDED VERBS ONLY)
13 TIG OF CDH (HRS,MIN,SEC)
14 CHECKLIST
   (USED BY EXTENDED VERBS ONLY)
15 INCREMENT MACHINE ADDRESS
16 TIME OF EVENT (HRS,MIN,SEC)
17 SPARE
18 AUTO MANEUVER BALL ANGLES
19 SPARE
20 ICDU ANGLES
21 PIPAS
22 NEW ICDU ANGLES
23 SPARE
24 DELTA TIME FOR AGC CLOCK(HRS,MIN,SEC)
25 CHECKLIST
   (USED WITH PLEASE PERFORM ONLY)
26 PRIO/DELAY, ADRES, BBCON
27 SELF TEST ON/OFF SWITCH
```

^
Wherever the text remains mostly black,
it's almost certainly correct!



Method 1 Continued ...

Proofing Magic: Colorize!

- 1) Don't use OCR to extract text.
- 2) Instead use OCR to overlay the transcribed text in *color* atop the scan.
- 3) Anywhere there's color after that ... double check!

SFTEMP1 SFTEMP2 WITH THE DP SFINTAB ENTRIES..

25 S C O M M A N D S IN SFTEMP1.

NN NORMAL NOUNS
00 NOT IN USE
01 SPECIFY MACHINE ADDRESS (FRACTIONAL)
02 SPECIFY MACHINE ADDRESS (WHOLE)
03 SPECIFY MACHINE ADDRESS (DEGREES)
04 ANGULAR ERROR/DIFFERENCE
05 ANGULAR ERROR/DIFFERENCE
06 OPTION CODE
07 ECADR OF WORD TO BE MODIFIED
08 ALARM DATA
09 ALARM CODES
10 CHANNEL TO BE SPECIFIED
11 TIG OF CSI (HRS,MIN,SEC)
12 OPTION CODE
13 TIG OF CDH (EXTENDED VERBS ONLY)
14 CHECKLIST
15 INCREMENT MACHINE ADDRESS
16 TIME OF EVENT (HRS,MIN,SEC)
17 SPARE
18 AUTO MANEUVER BALL ANGLES
19 SPARE
20 ICDU ANGLES
21 PIPAS
22 NEW ICDU ANGLES
23 SPARE
24 DELTA TIME FOR AGC CLOCK(HRS,MIN,SEC)
25 CHECKLIST
26 PRIO/DELAY, ADRES, BBCON
27 SELF TEST ON/OFF SWITCH

^
But ... wherever there's noticeable color,
there MAY be a transcription error.
In other words, you just
double-check stuff that's in color!
By the way, up close, the colors visually
POP OUT more than it may seem in these slides.



Method 1 Continued ...

Proofing Magic: Colorize!

- 1) Don't use OCR to extract text.
- 2) Instead use OCR to overlay the transcribed text in *color* atop the scan.
- 3) Anywhere there's color after that ... double check!

SFTEMP1 SFTEMP2 WITH THE DP SFINTAB ENTRIES..

25 S C U M M A T I E S I N S F T E M P 1 .

NN NORMAL NOUNS
00 NOT IN USE
01 SPECIFY MACHINE ADDRESS (FRACTIONAL)
02 SPECIFY MACHINE ADDRESS (WHOLE)
03 SPECIFY MACHINE ADDRESS (DEGREES)
04 ANGULAR ERROR/DIFFERENCE
05 ANGULAR ERROR/DIFFERENCE
06 OPTION CODE
07 ECADR OF WORD TO BE MODIFIED
08 ALARM DATA
09 ALARM CODES
10 CHANNEL TO BE SPECIFIED
11 TIG OF CSI (HRS,MIN,SEC)
12 OPTION CODE
13 TIG OF CDH (HRS,MIN,SEC)
14 CHECKLIST
15 INCREMENT MACHINE ADDRESS
16 TIME OF EVENT (HRS,MIN,SEC)
17 SPARE
18 AUTO MANEUVER BALL ANGLES
19 SPARE
20 ICDU ANGLES
21 PIPAS
22 NEW ICDU ANGLES
23 SPARE
24 DELTA TIME FOR AGC CLOCK(HRS,MIN,SEC)
25 CHECKLIST (USED WITH PLEASE PERFORM ONLY)
26 PRIO/DELAY ADRES, BBCON
27 SELF TEST ON/OFF SWITCH

^
The best example here is
the WITH at lower right.
It was PROBABLY incorrectly transcribed
as W T I H.

**Method 1 Continued ...
Proofing Magic: Compare!**

Page	Line	Code	Page	Line	Code
1185	BEE17	DEC 0	1185	BEE17	DEC 0
1185	D1/8	2DEC 1.0 B-3	1185	D1/8	2DEC 1.0 B-3
1185	D1/128	2DEC 1.0 B-7	1185	D1/128	2DEC 1.0 B-7
1185	D1/64	2DEC 1.0 B-6	1185	D1/64	2DEC 1.0 B-6
1185	D1/4	2DEC 1.0 B-2	1185	D1/4	2DEC 1.0 B-2
1185	D1/16	2DEC 1.0 B-4	1185	D1/16	2DEC 1.0 B-4
1185	D1/32	2DEC 1.0 B-5	1185	D1/32	2DEC 1.0 B-5
1185	D1/1024	2DEC 1.0 B-10	1185	D1/1024	2DEC 1.0 B-10
1185	D1/256	2DEC 1.0 B-8	1185	D1/256	2DEC 1.0 B-8
1185	DP9/10	2DEC .9	1185	DP9/10	2DEC .9
1185	KEPZER0	EQUALS LOGZEROS	1185	KEPZER0	EQUALS LOGZEROS
1185	-59SC	2DEC -50.0 B-12	1185	-59SC	2DEC -50.0 B-12
1185	2PISC	2DEC 6.28318530 B-6	1185	2PISC	2DEC 6.28318530 B-6
1185	BEE19	EQUALS D1/32 -1 # 2DEC 1.0 B-19 (00000 01000)	1185	BEE19	EQUALS D1/32 -1 # 2DEC 1.0 B-19 (00000 01000)
1185	BEE22	EQUALS D1/256 -1 # 2DEC 1.0 B-22 (00000 00100)	1185	BEE22	EQUALS D1/256 -1 # 2DEC 1.0 B-22 (00000 00100)
1185	ONERIT	2DEC 1.0 B-28	1185	ONERIT	2DEC 1.0 B-28
1185	COGUP LIM	2DEC -.999511597	1185	COGUP LIM	2DEC -.999511597
1185	COGLOLIM	2DEC -.999511597	1185	COGLOLIM	2DEC -.999511597
1186	TIMETHET	STQ # PL AT 0	1186	TIMETHET	STQ # PL AT 0
1186	BOV	0	1186	BOV	0
1186	VLOAD	PDVL # SETUP FOR PARAM CALL PL AT 6	1186	VLOAD	PDVL # STEUP FOR PARAM CALL PL AT 6
1186	RVEC	VVEC	1186	RVEC	VVEC
1186	CALL		1186	CALL	
1186	BOV	PARAM CALL # PL AT 0	1186	BOV	PARAM CALL # PL AT 0
1186	COGADWFL	GETX	1186	COGADWFL	GETX
1186	COMMOUT	DLOAD BON XT	1186	COMMOUT	DLOAD BON XT

^
 Another thing ...
 When there's transcribed code for multiple similar versions of the software, there are tools for comparing two or more versions side by side. Of course, differences MAY be valid version-related changes. But they may also be transcription errors. Two independent software transcriptions, aren't likely to have transcription errors in identical locations.



Method 1 Continued ... Proofing Magic: Collate!

Original Symbol Table (1969)				Modern Symbol Table (2022)								
PIPASC 37,3077	389	1	385	PLENTY 25,3211	555	1	555	POSTNV E9,1520 = 128	2	386	387	
PIPASC E3,1453	109	1	326	PLUSIRE 21,3713	1509	1	1509	POSTAND 37,3734	1336	2	1335	
PIPASC E3,1453	109	1	109	PLUSK 27,2423	769			004029, F:	PIPASC 37,3077	004074, F:	PLENTY 25,3211	
PIPASC E3,1453	109	1	109	PMINE 34,2073	632	2	641	004030, E:	PIPASC E3,1453	004075, F:	PLUSFIRE 21,3713	
PIPASC E3,1453	109	1	109	PMINM 34,2103	632			004031, E:	PIPASC E3,1453	004076, F:	PLUSK 27,2423	
PIPASC E3,1453	109	1	109	PMASK 10,3514	1372	1	1360	004032, F:	PIPASC 37,3077	004077, F:	PMINE 34,2073	
PIPASC E3,1453	109	1	109	POINTER 0156 = 1308	7	1306	1308	004033, F:	PIPASC 37,3077	004078, F:	PMINM 34,2103	
PIPASC E3,1453	109	1	109	POINTVSM E7,1172 = 146	7	152	956	004034, F:	PIPASC 37,3077	004079, F:	PMASK 10,3514	
PIPASC E3,1453	109	1	109	POLISH 0117 = 96	39	96	1094	004035, E:	PIPASC 37,3077	004080, E:	POINTER 0156	
PIPASC E3,1453	109	1	109	POLLEY 5051	1102	2	1102	004081, E:	PIPASC 37,3077	004081, E:	POINTVSM E7,1172	
PIPASC E3,1453	109	1	109	POLY 7222	1034	7	856	1282	004082, E:	PIPASC 37,3077	004082, E:	POLISH 0117
PIPASC E3,1453	109	1	109	POLYCOEF 0140 = 97	4	1034	1035	004083, E:	PIPASC 37,3077	004083, E:	POLLEY 5051	
PIPASC E3,1453	109	1	109	POLYCOJEF 12,3031	1189	1	1191	004084, F:	POLY 7222	004084, F:	POLY 7222	
PIPASC E3,1453	109	1	109	POLYCOJHM 7232	1034	1	1034	004085, E:	POLYCOEF 12,3031	004085, E:	POLYCOM 0140	
PIPASC E3,1453	109	1	109	POLYCOJHM 7242	1035	1	1035	004086, F:	POLYCOJEF 12,3031	004086, F:	POLYCOEF 12,3031	
PIPASC E3,1453	109	1	109	POLYCOJHM 0441 = 97	7	1034	1035	004087, F:	POLYCOJHM 7232	004087, F:	POLYCOM 0140	
PIPASC E3,1453	109	1	109	POLYCOJHM E6,1161 = 131	10	1430	1438	004088, F:	POLYCOJHM 7242	004088, F:	POLYCOM 0140	
PIPASC E3,1453	109	1	109	POLYCOJHM 7245	1035	1	1035	004089, E:	POLYCOJHM 7245	004089, E:	POLYRET 0141	
PIPASC E3,1453	109	1	109	POLYCOJHM 37,2251	376	1	376	004090, E:	POLYCOJHM 7245	004090, E:	POLYTEMP E6,1741	
PIPASC E3,1453	109	1	109	POLYCOJHM 37,2240	376			004091, F:	POLYCOJHM 7245	004091, F:	POLY2 7245	
PIPASC E3,1453	109	1	109	POLYCOJHM 37,2245	376	1	376	004092, F:	POLYCOJHM 7245	004092, F:	PON 37,2251	
PIPASC E3,1453	109	1	109	POLYCOJHM 5652	1383	3	244	1360	004093, F:	PON2 37,2240	004093, F:	PON2 37,2240
PIPASC E3,1453	109	1	109	POLYCOJHM 5726	1384	2	1079	1132	004094, F:	PON4 37,2245	004094, F:	PON4 37,2245
PIPASC E3,1453	109	1	109	POLYCOJHM 04,2257	231	1	230	004095, F:	PON4 37,2245	004095, F:	PODD00 5652	
PIPASC E3,1453	109	1	109	POLYCOJHM 04,2163	229			004096, F:	PON4 37,2245	004096, F:	PODD00 5726	
PIPASC E3,1453	109	1	109	POLYCOJHM 05,2547	214	1	230	004097, F:	PON4 37,2245	004097, F:	PODD01 5726	
PIPASC E3,1453	109	1	109	POLYCOJHM 33,3703	895	2	895	004098, F:	PON4 37,2245	004098, F:	POOH 04,2257	
PIPASC E3,1453	109	1	109	POLYCOJHM 21,3371	1479	2	1479	004099, F:	PON4 37,2245	004099, F:	POOH 04,2163	
PIPASC E3,1453	109	1	109	POLYCOJHM 21,3673	1509	1	1509	004100, F:	PON4 37,2245	004100, F:	POOH 05,2647	
PIPASC E3,1453	109	1	109	POLYCOJHM 12,2620	1183	1	1182	004101, F:	PON4 37,2245	004101, F:	POOH 33,3703	
PIPASC E3,1453	109	1	109	POLYCOJHM 12,2246	1177	1	1176	004102, F:	PON4 37,2245	004102, F:	POOH 21,3371	
PIPASC E3,1453	109	1	109	POLYCOJHM 21,3575	1483	2	1473	1483	004103, F:	PON4 37,2245	004103, F:	POOH 21,3673
PIPASC E3,1453	109	1	109	POLYCOJHM 42,3414	423	2	423	004104, F:	PON4 37,2245	004104, F:	POOH 33,3703	
PIPASC E3,1453	109	1	109	POLYCOJHM 15,3700	985	1	986	004105, F:	PON4 37,2245	004105, F:	POOH 21,3371	
PIPASC E3,1453	109	1	109	POLYCOJHM 15,3703	985	1	986	004106, F:	PON4 37,2245	004106, F:	POOH 21,3673	
PIPASC E3,1453	109	1	109	POLYCOJHM 15,3722	986	1	987	004107, F:	PON4 37,2245	004107, F:	POOH 12,242	
PIPASC E3,1453	109	1	109	POLYCOJHM 15,3770	987	1	986	004108, F:	PON4 37,2245	004108, F:	POOH 12,2620	
PIPASC E3,1453	109	1	109	POLYCOJHM 15,3743	985			004109, F:	PON4 37,2245	004109, F:	POOH 12,2246	
PIPASC E3,1453	109	1	109	POLYCOJHM 15,1416 = 127	6	127	381	004110, F:	PON4 37,2245	004110, F:	POOH 21,3575	
PIPASC E3,1453	109	1	109	POLYCOJHM 4733	1095	46	171	1506	004111, F:	PON4 37,2245	004111, F:	POOH 42,3414
PIPASC E3,1453	109	1	109	POLYCOJHM 4733	1095	46	171	1506	004112, F:	PON4 37,2245	004112, F:	POOH 12,2620
PIPASC E3,1453	109	1	109	POLYCOJHM 4733	1095	46	171	1506	004113, F:	PON4 37,2245	004113, F:	POOH 12,2246
PIPASC E3,1453	109	1	109	POLYCOJHM 4733	1095	46	171	1506	004114, F:	PON4 37,2245	004114, F:	POOH 21,3575
PIPASC E3,1453	109	1	109	POLYCOJHM 4733	1095	46	171	1506	004115, F:	PON4 37,2245	004115, F:	POOH 21,3575
PIPASC E3,1453	109	1	109	POLYCOJHM 4733	1095	46	171	1506	004116, F:	PON4 37,2245	004116, F:	POOH 21,3575
PIPASC E3,1453	109	1	109	POLYCOJHM 4733	1095	46	171	1506	004117, F:	PON4 37,2245	004117, F:	POOH 21,3575
PIPASC E3,1453	109	1	109	POLYCOJHM 4733	1095	46	171	1506	004118, F:	PON4 37,2245	004118, F:	POOH 21,3575

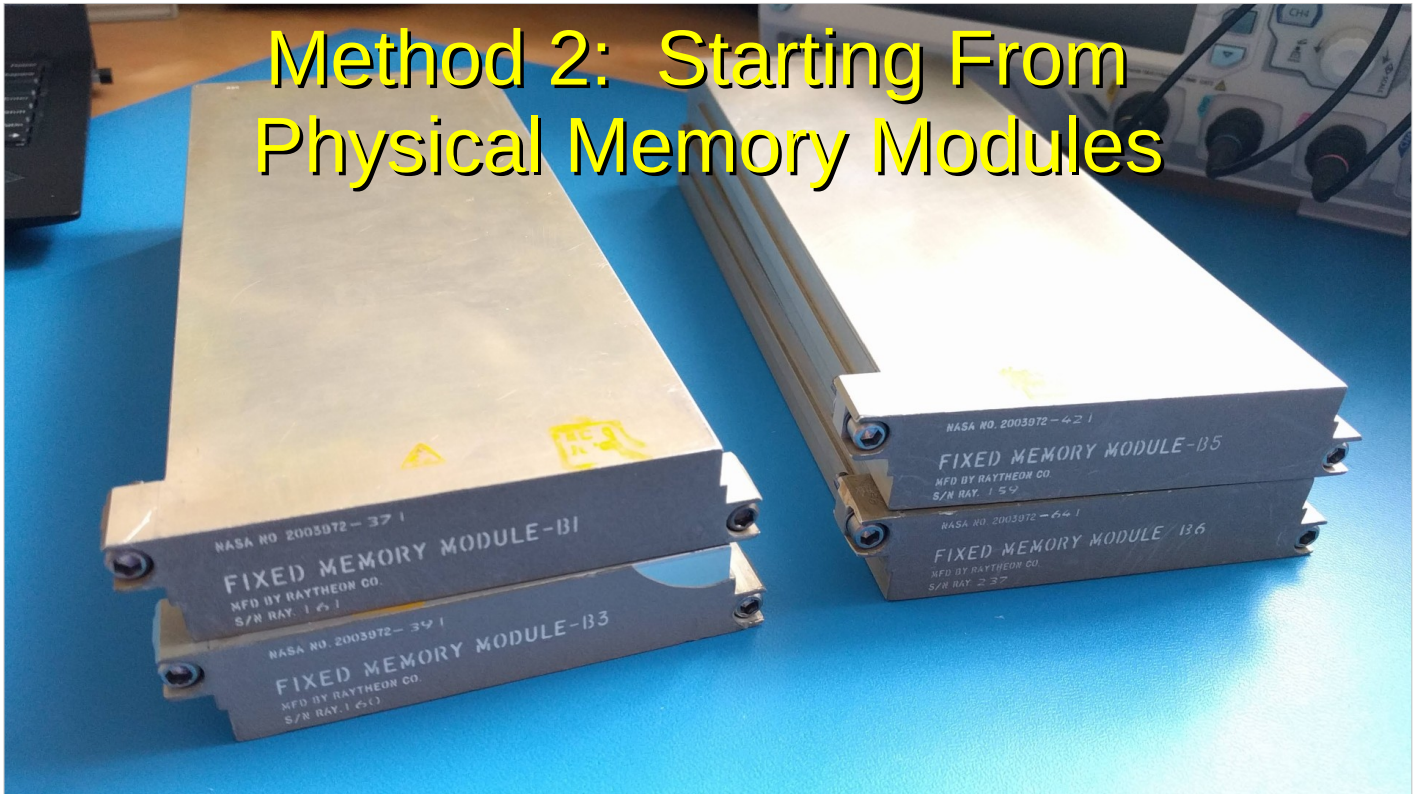
^

Yet another sometimes-difficult-to-detect problem is when a symbol is misspelled CONSISTENTLY throughout an entire transcription. OUR stereotypical problem is the symbol "POOH" and its friends. Is it P-Oh-Oh-H or is it P-Zero-Zero-H? Comparing the sorted symbol table from the original assembly listing versus the one output by our modern assembler is a powerful way to find such misspelled symbols ...

... IF the layouts of the tables and the sorting orders of the symbols are the same. Which can be a very tricky problem!

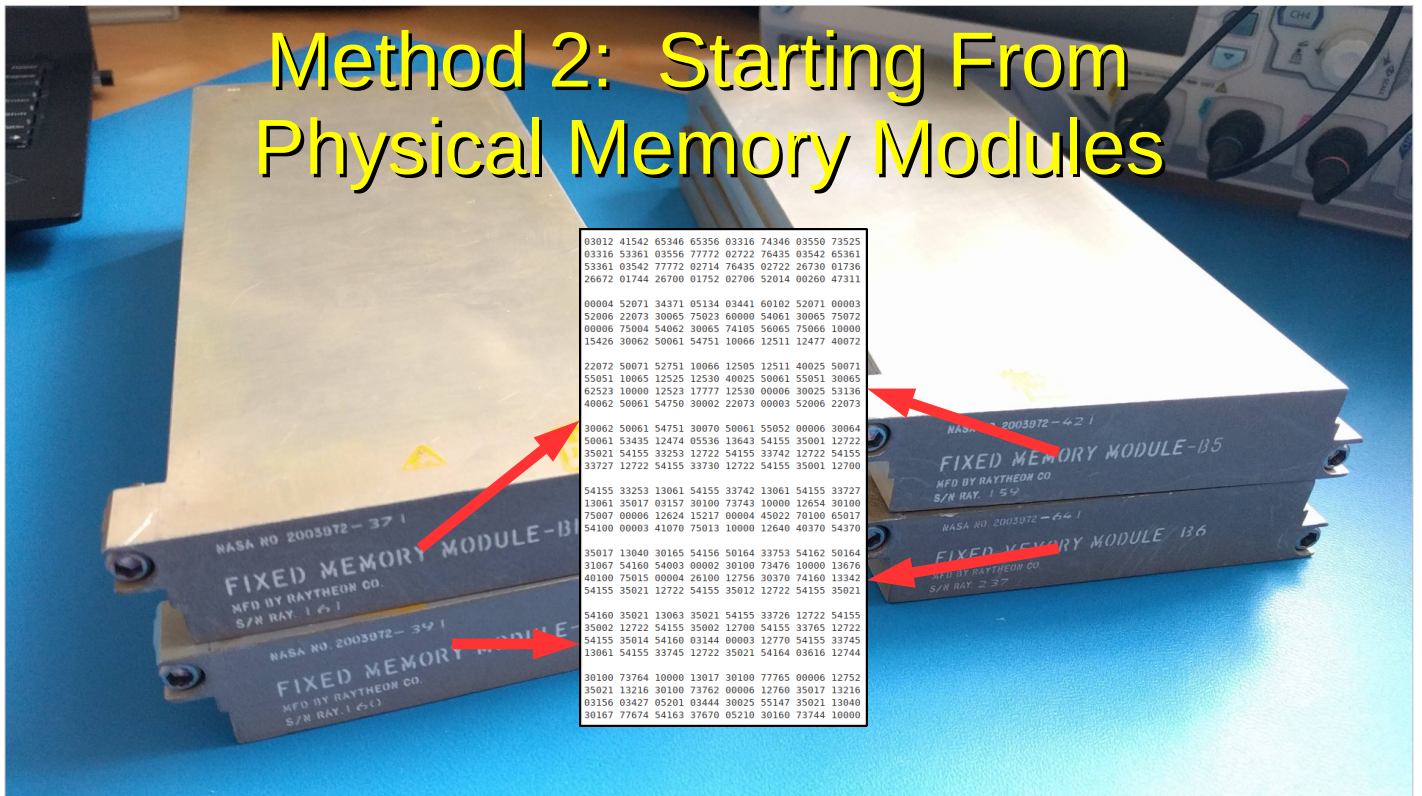
Now ... recall that I said we have three very different methods of acquiring AGC source code. That's all I have to say about the 1st method, so let's move on to the 2nd.

Method 2: Starting From Physical Memory Modules



^
Suppose you DON'T have an Apollo-era printout to work with. Sometimes museums or collectors own physical AGC rope memory modules. These are cartridges of read-only memory, up to 6 of them per AGC, that hold the executable software.

Method 2: Starting From Physical Memory Modules



^
Sometimes those owners can be persuaded to give us access to them. We're able to dump the contents of the modules, creating an octal listing of the executable. The modules have built-in parity bits and check-sums, for extra confidence that dumps are valid.

Method 2: Starting From Physical Memory Modules

```

03012 41542 65346 65356 03316 74346 03550 73525
03316 53361 03556 77772 02722 76435 03542 65361
53361 03542 77772 02714 76435 02722 26730 01736
26672 01744 26700 01752 02706 52014 00260 47311

00004 52071 34371 05134 03441 60102 52071 00003
52006 22073 30065 75023 06000 54061 30065 75072
00006 75004 54062 30065 74105 56065 75066 10000
15426 30062 50061 54751 10066 12511 12477 40072

22072 50071 52751 10066 12505 12511 40025 50071
55051 10065 12525 12530 40025 50061 55051 30065
62523 10000 12523 17777 12530 00006 30025 53136
40062 50061 54750 30002 22073 00005 52006 22073

30062 50061 54751 30070 50061 55052 00006 30064
50061 53435 12474 05536 13643 54155 35001 12722
35021 54155 33253 12722 54155 33742 12722 54155
33727 12722 54155 33730 12722 54155 35001 12700

54155 33253 13061 54155 33742 13061 54155 33727
13061 35017 03157 30100 73743 10000 12654 30100
75007 00006 12624 15217 00004 45022 70100 65017
54100 00003 41070 75013 10000 12640 40370 54370

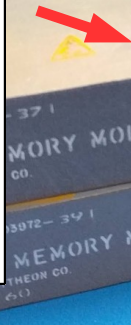
35017 13040 30165 54156 50164 33753 54162 50164
31067 54160 54003 00002 30100 73476 10000 13676
40100 75015 00004 26100 12756 30370 74160 13342
54155 35021 12722 54155 35012 12722 54155 35021

54160 35021 13063 35021 54155 33726 12722 54155
35002 12722 54155 35002 12700 54155 33765 12722
54155 35014 54160 03144 00003 12770 54155 33744
13061 54155 33745 12722 35021 54164 03610 12744

30100 73764 10000 13017 30100 77765 00006 12752
35021 13216 30100 73762 00006 12760 35017 13216
03156 03427 05201 03444 30025 55147 35021 13040
30167 77674 54163 37670 05210 30160 73744 10000
  
```

```

02,0000 00004 INHINT
02,0001 34054 CA 4054
02,0002 56006 XCH 0006
02,0003 12072 TCF 2072
02,0004 52011 DXCH 0010
02,0005 00006 EXTEND
02,0006 22012 QXCH 0012
02,0007 14175 TCF 4175
02,0010 52011 DXCH 0010
02,0011 00006 EXTEND
02,0012 30346 DCA 0345
02,0013 52006 DXCH 0005
02,0014 52011 DXCH 0010
02,0015 34056 CA 4056
02,0016 56006 XCH 0006
02,0017 12736 TCF 2736
02,0020 52011 DXCH 0010
02,0021 37605 CA 7605
02,0022 14713 TCF 4713
02,0023 24006 INCR 0006
02,0024 52011 DXCH 0010
02,0025 34057 CA 4057
02,0026 56006 XCH 0006
02,0027 12362 TCF 2362
02,0030 52011 DXCH 0010
02,0031 34060 CA 4060
02,0032 56006 XCH 0006
02,0033 13161 TCF 3161
02,0034 52011 DXCH 0010
02,0035 34057 CA 4057
02,0036 56006 XCH 0006
02,0037 12402 TCF 2402
02,0040 52011 DXCH 0010
02,0041 34061 CA 4061
02,0042 56006 XCH 0006
02,0043 12047 TCF 2047
02,0044 52011 DXCH 0010
02,0045 34062 CA 4062
02,0046 56006 XCH 0006
  
```



^
 We can disassemble such a dumped octal listing to get rough, imperfect source code for that AGC software version.

Method 2: Starting From Physical Memory Modules

```

03012 41542 65346 65356 03316 74346 03550 73525
03316 53361 03556 77772 02722 76435 03542 65361
53361 03542 77772 02714 76435 02722 26730 01736
26672 01744 26700 01752 02706 52014 00260 47311

00004 52071 34371 05134 03441 60102 52071 00003
52006 22073 30065 75023 60000 54061 30065 75072
00006 75004 54062 30065 74105 56065 75066 10000
15426 30062 50061 54751 10066 12511 12477 40072

22072 50071 52751 10066 12505 12511 40025 50071
55051 10065 12525 12530 40025 50061 55051 30065
62523 10000 12523 17777 12530 00006 30025 53136
40062 50061 54750 30002 22073 00005 52006 22073

30062 50061 54751 30070 50061 55052 00006 30064
50061 53435 12474 05536 13643 54155 35001 12722
35021 54155 33253 12722 54155 33742 12722 54155
33727 12722 54155 33730 12722 54155 35001 12700

54155 33253 13061 54155 33742 13061 54155 33727
13061 35017 03157 30100 73743 10000 12654 30100
75007 00006 12624 15217 00004 45022 70100 65017
54100 00003 41070 75013 10000 12640 40370 54370

35017 13040 30165 54156 50164 33753 54162 50164
31067 54160 54003 00002 30100 73476 10000 13076
40100 75015 00004 26100 12756 30370 74160 13342
54155 35021 12722 54155 35012 12722 54155 35021

54160 35021 13063 35021 54155 33726 12722 54155
35002 12722 54155 35002 12700 54155 33765 12722
54155 35014 54160 03144 00003 12770 54155 33745
13061 54155 33745 12722 35021 54164 03610 12744

30100 73764 10000 13017 30100 77765 00006 12752
35021 13216 30100 73762 00006 12760 35017 13216
03156 03427 05201 03444 30025 55147 35021 13040
30167 77674 54163 37670 05210 30160 73744 10000
  
```

```

02,0000 00004 INHINT 4054
02,0001 34054 CA 4054
02,0002 56006 XCH 0006
02,0003 12072 TCF 2072
02,0004 52011 DXCH 0010
02,0005 00006 EXTEND 0012
02,0006 22012 QXCH 4175
02,0007 14175 TCF 0010
02,0010 52011 DXCH 0010
02,0011 00006 EXTEND 0010
02,0012 30346 DCA 0345
02,0013 52006 DXCH 0005
02,0014 52011 DXCH 0010
02,0015 34056 CA 4056
02,0016 56006 XCH 0006
02,0017 12736 TCF 2736
02,0020 52011 DXCH 0010
02,0021 37665 CA 7665
02,0022 14713 TCF 4713
02,0023 24006 INCR 0006
02,0024 54011 DXCH 0010
02,0025 34057 CA 4057
02,0026 56006 XCH 0006
02,0027 12362 TCF 2362
02,0030 52011 DXCH 0010
02,0031 34060 CA 4060
02,0032 56006 XCH 0006
02,0033 13161 TCF 3161
02,0034 52011 DXCH 0010
02,0035 34057 CA 4057
02,0036 56006 XCH 0006
02,0037 12402 TCF 2402
02,0040 52011 DXCH 0010
02,0041 34061 CA 4061
02,0042 56006 XCH 0006
02,0043 12047 TCF 2047
02,0044 52011 DXCH 0010
02,0045 34062 CA 4062
02,0046 56006 XCH 0006
  
```

```

SETLOC 4000
INXINT # GO
CAP G000
XCH SBANK
TCF GOPROG

DXCH ARUPT # HERE ON A TRUPT
EXTEND
QXCH ORUPT
TCF DOTRUPT IS IN FIX-FIXED.(INTR-BANK COM)

DXCH ARUPT # TSRUPT
EXTEND
DCA TSLOC # TSLOC EQUALS TSADR
DCB

DXCH ARUPT # TSRUPT
CAP T3RUPTBB
XCH SBANK
TCF T3RUPT

DXCH ARUPT # T4RUPT
CAP ZERO
TCF T4RUPT
EBANK-
BBCON T4RUPTA

DXCH ARUPT # KEYRUPT1
CAP KEYRUPTBB
XCH SBANK
TCF KEYRUPT1

DXCH ARUPT # KEYRUPT2
CAP MKRUPTBB
XCH SBANK
TCF MKRUPT

DXCH ARUPT # UPRUPT
CAP UPRUPTBB
XCH SBANK
TCF UPRUPT

DXCH ARUPT # DOWNRUPT
CAP DNWRUPTBB
XCH SBANK
TCF DODOWNTM

DXCH ARUPT # RADAR RUPT
CAP R0RUPTBB
  
```

^
 Finally, the imperfect source code often can be perfected by comparing it to similar software versions and importing chunks of source code from those similar versions.



Method 3: Starting From Software-Change Paper Trail

Requirements:

- Program listing(s) for very similar versions; *and*
- Complete documentation of changes; *and*
- Knowledge of the expected checksums.

^

But what if there's NEITHER a printout
NOR a physical memory module?

Under the rare conditions listed in this slide,
an AGC software version can
SOMETIMES be reconstructed anyway.

First, you clone the source code for
the software version that you THINK
is closest to the one you want to reconstruct.

Then, one-by-one, you edit in EACH of
the software changes that are described
in the Apollo-era paper trail.

Mostly, that means pasting code from
a similar AGC version in which you know
that the SAME change had also been made.

Having done all that,
you assemble the edited code.

If the check-sums are as hoped-for,
then success!

For example, the Apollo 10 Command Module
software and the Apollo 14 LEM software
were reconstructed in exactly this way.

As a final check,
you fly the mission in a spaceflight simulator.



Apollo 15 – Lunar Apennines



^
Speaking of which, our CPU emulator and our collected AGC software have been integrated into the ORBITER spaceflight-simulation system with the NASSP add-on.
That's N A S S P.
So in Orbiter, Apollo missions use a fully operational AGC.
We're looking at a simulated Apollo 15 lunar landing at Hadley Rille.
The landing is just under 14 minutes total, but we're only going to see a bit from the middle, as the LEM swoops down over the Lunar Apennine Mountains.
For this mission, the LEM's AGC runs software known as LUMINARY 210, and specifically it's using the subprograms called P63, P64, and P66. At the moment, P63, "landing maneuver braking phase", is running.
Incidentally, the simulation does have a human pilot, Nik.
But it so happens that the AGC can handle this lunar landing automatically, so Nik is basically just observing the action the same way we are, and occasionally checking the DSKY at the bottom of the control panel.
The P63 program is now nearing its end, after which it will automatically transition to program P64, "landing maneuver approach phase", at which point the LEM will pitch over. But I'm going to end the simulation when that happens, so we'll miss the rest of the landing ... and there's pitch-over.



Virtual AGC Project Achievements Over 2 Decades

<u>Item</u>	<u>2003</u>	<u>2022</u>
Documents	~100	~2800
AGC software versions	2	~33
Fully covered missions (AGC)	None!	Apollo 4, 5, 6, 8, 9, 10, 11, 15, 16, 17
Partially covered missions (AGC)	Apollo 9, 13	Apollo 12, 13, 14
Spaceflight simulation systems	None!	AGC and AGS integrated into Orbiter with NASSP add-on
Electrical and mechanical drawings	0	~100K scans
AGS software versions	0	2 (Apollo 11, 12, 15, 16, 17)
LVDC software versions	0	1 (Not complete, not fully developed, mission not flown)
Gemini OBC software versions	0	Debatable
Wish list	n/a	Apollo 1 AGC, Apollo 7 AGC, LVDC, Gemini OBC, ITAR!

^
A couple of decades has made a big difference in the amount of publicly available Apollo flight software and related material. 2 AGC software versions have turned into over 30, representing the full AGC software needed for 10 different Apollo missions. And a couple more missions seem to be on the way as I speak! Meanwhile, a mere handful of documents has turned into nearly 3000. But the wish list does still contain some very-significant items. Now ... that's all I have about the AGC today, but I do have some time left over



^

so let me tell you a story about
the Launch Vehicle Digital Computer, or LVDC.
This is yet another Apollo flight computer
covered by our project,
but different from the AGC.
The story will illustrate a few of the kinds of
preservation-related frustrations we experience.
Realize that the AGC itself couldn't control
the actions of the Saturn V rocket, mostly.
That job was performed by the LVDC,
developed and manufactured by
IBM's Federal Services Division.



A Launch Vehicle Digital Computer Story

- Years-long search for code.
- U. S. Space & Rocket Center archive told me:
 - They had a copy of the software
 - Because of contract with IBM I couldn't scan it.
 - But I could transcribe it to paper by hand!



^
At some point, we got word that the U. S. Space & Rocket Center Archive had a copy of the software! But the archive told me that they couldn't legally allow me to digitize the program listing. If I wanted it, they said, I'd have to copy it by hand. Which sounded pretty unpleasant, but eventually I decided to do it anyway. So, on my vacation I drove 750 miles from Dallas, Texas to Huntsville, Alabama.



A Launch Vehicle Digital Computer Story

- Years-long search for code
- U. S. Space & Rocket Center archive told me:
 - They had a copy of the software ... FALSE
 - Because of contract with IBM I couldn't scan it.
 - But I could transcribe it to paper by hand!



^
Only to find, once I got there,
that it was a mix-up by the archive!
In fact, they had no LVDC software at all.
Since I happened to be there anyway,
I talked to some of the Museum's docents
who were Saturn V old-timers.
One of those was an IBM Federal Services
Division manager for LVDC software development,
and he told me some things.



A Launch Vehicle Digital Computer Story

- Years-long search for code
- U. S. Space & Rocket Center archive told me:
 - They had a copy of the software ... FALSE
 - Because of contract with IBM I couldn't scan it.
 - But I could transcribe it to paper by hand!
- IBM FSD software manager told me:
 - Nobody could possibly have a copy
 - Software was classified
 - Was destroyed after each mission
 - Would be pointless to run in an emulator



^
Nobody could possibly have a copy of the code,
he said, because it was classified.
You didn't just take classified material
home with you.
Also, he insisted, the source code
was destroyed after each mission.
And finally, he said, my project
was completely worthless,
because the LVDC could never be run
as a simulation anyway without simulating
certain of its peripheral devices too,
something he imagined was impossible.



A Launch Vehicle Digital Computer Story

- Years-long search for code
- U. S. Space & Rocket Center archive told me:
 - They had a copy of the software ... FALSE
 - Because of contract with IBM I couldn't scan it.
 - But I could transcribe it to paper by hand!
- IBM FSD software manager told me:
 - Nobody could possibly have a copy ... FALSE
 - Software was classified ... FALSE
 - Was destroyed after each mission ... FALSE
 - Would be pointless to run in an emulator ... ?

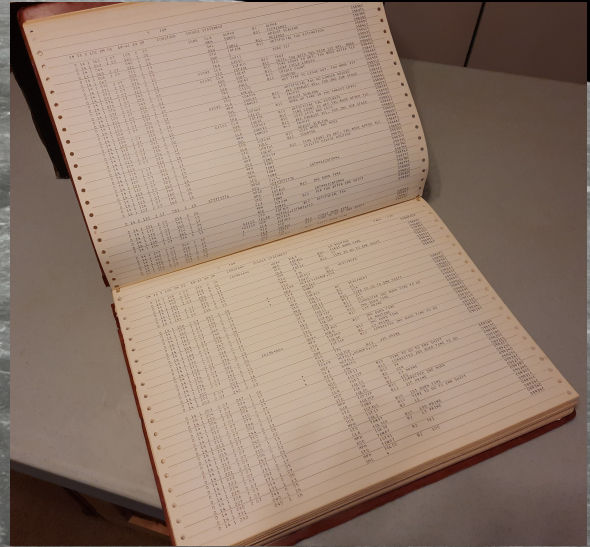


^
I know NOW that these claims were mostly false,
because ...



A Launch Vehicle Digital Computer Story

- Years-long search for code
- U. S. Space & Rocket Center archive told me:
 - They had a copy of the software ... FALSE
 - Because of contract with IBM I couldn't scan it.
 - But I could transcribe it to paper by hand!
- IBM FSD software manager told me:
 - Nobody could possibly have a copy ... FALSE
 - Software was classified ... FALSE
 - Was destroyed after each mission ... FALSE
 - Would be pointless to run in an emulator ... ?
- Original developer turns into a White Knight!

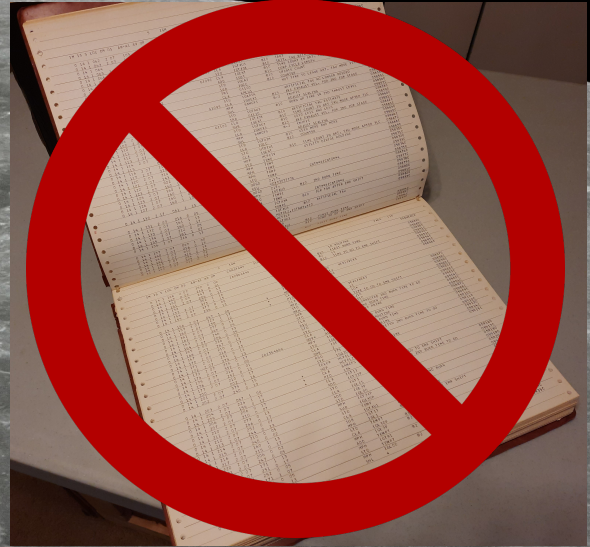


^
a few days before the 50th anniversary
of the Apollo 11 landing,
one of original LVDC developers
unexpectedly appeared and
gave me a printout of an LVDC assembly listing.
Admittedly, this was a buggy engineering
version of the code,
for an alternative mission profile
that never flew.
But nevertheless, it's a big, fat printout
of real LVDC source code.
And no, it's NOT marked as being classified.



A Launch Vehicle Digital Computer Story

- Years-long search for code
- U. S. Space & Rocket Center archive told me:
 - They had a copy of the software ... FALSE
 - Because of contract with IBM I couldn't scan it.
 - But I could transcribe it to paper by hand!
- IBM FSD software manager told me:
 - Nobody could possibly have a copy ... FALSE
 - Software was classified ... FALSE
 - Was destroyed after each mission ... FALSE
 - Would be pointless to run in an emulator ... ?
- Original developer turns into a White Knight!
- International Traffic in Arms Regulations (ITAR)
 - Is it legal to post publicly online???



^

But ... I've NOT posted the source code online,
and I WON'T give you a copy of it
unless you can prove to me that you're
legally a "U. S. Person".

Why?

You see, in theory a Saturn V could be used
as a launch vehicle to deliver a warhead ...
if you had a billion dollars to build the rocket
and a launch facility that could handle it.
Under that theory, "export" of the software
MAY be prohibited due to U. S. regulations
known as the International Traffic in Arms
Regulations, or ITAR.

I've had attorneys specializing in space law
looking at the question of whether or not ITAR
actually applies to LVDC source code or not.
They've been looking at it for 3 years.



Some Lessons Learned

- 1) There's no such thing as *too much* documentation.
- 2) Nor *too much* code.
- 3) More of the past may be recoverable than you think.
- 4) Unexpected help may be just around the corner.
- 5) Things that presumed experts tell you are not always correct.
- 6) Human relations may be more helpful than online resources.
- 7) Expect obstruction, indifference ... and occasional cooperation.

^
Finally ... here are a few takeaways ...
platitudes, really ... from my experiences
with the Virtual AGC Project.
If I had to choose just one,
I think I'd pick number 3 -
More software from the past
may be recoverable than you think.



Some Lessons Learned

- 1) There's no such thing as *too much* documentation.
- 2) Nor *too much* code.
- 3) More of the past may be recoverable than you think.
- 4) Unexpected help may be just around the corner.
- 5) Things that presumed experts tell you are not always correct.
- 6) Human relations may be more helpful than online resources.
- 7) Expect obstruction, indifference ... and occasional cooperation.

^
But never forget that software recovery
may ALSO be dependent on the extra material
you collect, even if it may not itself be
software or may at first seem irrelevant.