# MCR10130
# Fix Issue with GCOS TSS and GCOS Batch Simulators
# Revision 1.2

Eric Swenson

December 9, 2022

## 1 Problem Description

Two problems have recently been discovered with GCOS processing. The first it with the GCOS TSS simulator, and the second is more general, with the GCOS simulator.

Ticket https://multics-trac.swenson.org/ticket/291 describes the first problem, which prevents the Jovial or Algol compilers from running under TSS.

It is not possible to run the Jovial or Algol compilers within the GTSS subsystem on Multics due to memory size limits being too small. As a result, programs terminate with the following error:

```
m4/n4-i/o lim. call/save
 *ABT  0003t-01 @1043.3 m4/n4-i/o lim. call/save
```

Increasing the memory limit allows the Jovial (and Algol) compilers to run.

Ticket https://multics-trac.swenson.org/ticket/299 describes the second problem, with the GCOS batch simulator.

The GCOS batch simulator begins with making a first pass over the card deck to determine, among other things, whether there is an EXECUTE card in the deck. A deck with an upper case EXECUTE card (which is perfectly legitimate, and common in GCOS days) causes the first pass to fail to find the EXECUTE card, because it compares card types with a table whose names are all in lowercase. Other cards that are searched for in the first pass are also not found if they are in upper case.

The main processing loop of the GCOS simulator, on the other hand, does convert cards to lowercase before comparing them with a table of cards in order to know how to process the card deck.

## 2 Proposed Fix

The proposed fix to the memory limit problem is simply to increase the memory size limit to a large enough value that a JOVIAL or Algol compilation is allowed to run. The new limit was determined through experimentation and gradual increasing until the smallest value that supported the JOVIAL compiler was found.

The limits are specified in the program gcos_control_tables_.alm using an ALM macro to define limits for a specific card thus:

```
define   progra,??????,,i*,5,17,5,dollar_t
```

While there are "jovial" and "algol" cards, in addition to the "progra" card, experimentation indicates that it is the "progra" card memory size limit that is preventing the JOVIAL compiler from running. Increasing only this value allows the JOVIAL (and Algol) compilers to run.

The 17 for the "progra" card, shown above, specifies the memory limit of 17K words for the "progra" card. Increasing this value to 29K words allows the JOVIAL (and Algol) compilers to run.

While it has not been shown, yet, that the cpu limit for the "progra" card, specified in units of hundredths of cpu hours, this MCR also proposes increasing the cpu limit, based on other values found in documentation, and the values specified for the "jovial" and "algol" cards. We propose to increase the cpu limit from 5 hundredths of a cpu hour to 8. An MCR reviewer recommended this specific value.

While increasing the value of the "progra" card, as opposed to the "jovial" (or "algol") card seems a bit counter-intuitive, it is suspected that the those cards apply to the *running* of a JOVIAL (or Algol) program, as opposed to the *compiling* of it.

Since we have not yet managed to construct a JOVIAL program that compiles, we don't know yet whether we need to increase any limits on the JOVIAL card. Therefore until we do, this MCR doesn't propose increasing any other limits on JOVIAL or ALGOL cards, or any other.

A compare_ascii output showing the change is below:

```
cpa [lpn gcos_control_tables_.alm] ==

A277              define   progra,??????,,i*,5,17,5,dollar_t
Changed by B to:
B277              define   progra,??????,,i*,8,29,5,dollar_t

Comparison finished: 1 difference, 2 lines.

GHM 16:00 1.325 1 Swenson
```

The proposed fix to the second issue, that of the GCOS batch processor's comparing cards such that upper-case cards are not correctly handled, is simply to do what the main GCOS processing loop does – convert card types to lowercase. The main processing loop is in gcos_gein_.pl1. The first pass through the card deck is done in gcos_gein_pass1_.pl1.

A compare_ascii output showing that fix is shown below:

```
cpa [lpn gcos_gein_pass1_.pl1] ==

Inserted in B:
B785                  card_type = translate (card_type, LOWER_CASE, UPPER_CASE);
Preceding:
A785                  return;
```

```
Comparison finished: 1 difference, 1 line.
r 15:46 1.365 22
```

---

# 3 Documentation

No changes to documentation are necessary – as the changes are only internal or have never been documented.

# 4 Test

First, the gtss subsystem will be used to invoke the compiler as specified in the ticket, making sure that it runs rather than aborts due to limits. Then a few other gtss runs will be performed to make sure that everything continues to work.

For the GCOS batch processing problem, a few jobs that first compile and then execute programs will be run to make sure that after a successful compilation, the program is executed. We have two jobs at this point – one Algol and one Cobol as test cases.

# 5 Version History

| | | |
|---|---|---|
| 2022-12-07 | 1.0 | Initial version of the MCR. |
| 2022-12-08 | 1.1 | Increase limits further and correct issue with units. |
| 2022-12-09 | 1.2 | Fix case-comparison issue with GCOS batch processor. |