MCR10137
Update Hardcore to Report new PROM Fields and To Warn About Old Simulator
Revision 1.5

Eric Swenson

July 15, 2023

1 Problem Description

The simulator used to run Multics (DPS8M) has recently been updated to add additional fields to the simulated PROM. These new fields identify the version of the simulator that is running. They include major, minor, patch, and iteration version numbers, a human-readable version string, information about the build and target architecture and OS used when building the simulator. Finally, they include a field to specify the version of the PROM layout. That field, in hardware implementations of the DPS 8/M processors with ID PROMs, was always 255 (decimal), and has been set to 0, 1, and 2 (so far) in iterations of the ID PROM layout. All of these new fields (and most of the original fields) are simple fixed-length strings.

The ID PROM was only present on the DPS 8/M processors, and in fact, an option, so theoretically not even always present in those processors. The DPS8M simulator is faithful to the hardware implementation, and only populates the PROM when a CPU is configured as being a DPS 8/M CPU (not an L68 CPU).

On startup, Multics checks to see if the bootload processor is a DPS 8/M CPU and if so, whether it is configured with an ID PROM. If so, it reads the PROM and logs to the syserr log some of the fields (model number, serial number, and product ship date). This MCR recommends adding support to display the new fields. In addition, since it will now be possible (with the new PROM fields) to determine the version of the simulator, and various other information about the build of the simulator, this MCR will cause a warning to be issued on the operator console (and in the syserr log) when an old version of the simulator is used. This latter feature can be used in the future to warn about versions of the simulator that are known to cause issues with the proper running of Multics.

The "Multics Differences Manual DPS 8 70/M" (issued August 1983), defines the PROM fields. After correcting for some inaccuracies in this document's information, based on examination of the Multics code that reads the ID PROM, the following table describes the existing fields, along with their decimal offsets and lengths.

Offset	Length	Contents
0	11	CPU Model Number
11	11	CPU Serial Number
22	6	Date-Ship Code (YYMMDD)
28	5	CPU ID Field
40	1	Operating System Use
41	983	To be defined

The simulator was recently updated to include the following additional fields:

Offset	Length	Contents
60	1	PROM Layout Version
70	10	Release Git Commit Date (YYYY-MM-DD)
80	3	Release Major Version (NNN)
83	3	Release Minor Version (NNN)
86	3	Release Patch Version (NNN)
89	3	Release Iteration (NNN)
92	8	Build Number
100	1	Release Type
101	29	Release Version Text (e.g. 3.0.1-alpha1)
130	20	Build Architecture (e.g. x86_64)
150	20	Build Operating System (e.g. Linux)
170	20	Target Architecture (e.g. Intel x86_64 (AMD64))
190	20	Target Operating System (e.g. Windows)

The Release Type field will contain one of the following letters:

Value	Meaning
A	Alpha Release
B	Beta Release
C	Release Candidate
R	General Release
X	Unknown Release Type

The existing CPU Model Number can be used to determine whether Multics is running on hardware or the simulator – the simulator uses "DPS 8/SIM M" as the value of this field, which is distinct from that used by the hardware.

The PROM Layout Version field can be used to determine whether the new fields are present in the PROM or not. The value of this field, up until recent simulator versions (and including all hardware PROMs) has been 255 (decimal) or 377 (octal).

The existing code in `tc_init.pl1` emits a message like the following into the syserr log on startup:

```
CPU A: Model #: DPS 8/SIM M; Serial #: 1113; Ship date: 230702.
```

The existing CPU Model Number, CPU Serial Number, and Date-Ship Code ID PROM fields are used to populate the values in this message.

This MCR proposes to change this message (in cases where the PROM Layout Version is not 255 (decimal)) to one similar to this one:

```
CPU A: Model #: DPS 8/SIM M; Serial #: 1113; Ship date: 230709; PROM Layout Version: 2;
        Simulator Release: X3.0.1.1 (2023-07-09); Build Number: 12345678;
        Build Arch: x86_64; Build OS: Linux;
        Target Arch: Unknown Target Arch.; Target OS: Unknown Target OpSys.
```

with values provided by the corresponding new fields in the PROM. The Release Major, Minor, Patch, and Iteration Version fields will not be included in this message, but can be used by future versions of Multics to guard against running on old/incompatible simulators, or simulators with bugs known to cause issues when running Multics. The Simulator Release component of that message combines the Release Type field and the Release Version Text field, as is the convention used by the simulator when displaying its version.

In addition, this MCR proposes to add a warning such as the following:

```
Warning: You are running an old version of the simulator. Please update to R3.0.1 or
    later. See https://dps8m.gitlab.io/ for details.
```

when the simulator is found to NOT include the additional fields, or when the version of the simulator booting Multics is known to have issues. Note that despite the warning, Multics will boot regardless. In addition, this message is never emitted if the CPU Model Number represents hardware, rather than the DPS8M simulator.

The issue is reported in this ticket: https://multics-trac.swenson.org/ticket/294.

2 Proposed Fix

The proposed fix is to modify the Multics hardcore (in `tc_init.pl1` to emit a message of this form to the syserr log in place of the current message:

```
CPU A: Model #: DPS 8/SIM M; Serial #: 1113; Ship date: 230709; PROM Layout Version: 2;
        Simulator Release: X3.0.1.1 (2023-07-09); Build Number: 12345678;
        Build Arch: x86_64; Build OS: Linux;
        Target Arch: Unknown Target Arch.; Target OS: Unknown Target OpSys.
```

using values from the new fields in the PROM if they are present. The PROM Layout Version field will be used to decide whether the new fields are present. Just as is the case currently, the message is not emitted if the CPU is not a DPS 8/M processor, or if it is configured with no PROM present. The simulator configures all DPS 8/M simulated processors as having a PROM. If the PROM Layout Version field is found to be 255, no new fields will be read from the PROM and the CPU syserr message will remain as it is today.

The existing message that the above message replaces is only emitted to the syserr log, and not to the operator console. An MCR reviewer recommended that the message be also emitted to the console, so the implementation will do so.

In addition, `tc_init.pl1` will be updated to emit another syserr/console warning if the PROM is present, and the CPU Layout Version is 255. The warning message will look similar to this:

```
1035.0 Warning: You are running an old version of the DPS8M simulator.
```

Special handling of the "Build Number" is warranted. There are three cases:

- The simulator has computed a build number and advertises this in the ID PROM.
- The simulator has not computed a build number, but has filled the "Build Number" field in the ID PROM with spaces.
- The simulator does not advertise a build number, or does not populate the ID PROM.

In the first case, above, Multics will emit "Build Number: " followed by the value of the Build Number string from the ID PROM.

In the second case and third cases, Multics will emit: "Build Number: <None>". The special value will be clearly distinguishable from any real build number values.

The changes can be best seen from the `compare_ascii` output:

```
A88      dcl  cpu_model char (11) aligned;                    /* storage for cpu model
    number (from ID PROM) */
A89      dcl  cpu_serial char (11) aligned;                   /* storage for cpu serial
    number (from ID PROM) */
A90      dcl  cpu_ship_date char (6) aligned;                 /* storage for cpu ship
    date (from ID PROM) */
A91      dcl  table_value fixed bin;
A92
Changed by B to:
B88      dcl  cpu_model char (11) aligned;                /* cpu model number (offset 0 in
    ID PROM) */
B89      dcl  cpu_serial char (11) aligned;               /* cpu serial number (offset 11
    in ID PROM) */
B90      dcl  cpu_ship_date char (6) aligned;             /* cpu ship date (offset 22 in
    ID PROM) */
B91      dcl  prom_layout_version   char (1) aligned;      /* emu layout version (offfset
    60 in ID PROM) */
B92      dcl  prom_commit_date      char (10) aligned;     /* emu commit date (offset 70
    in ID PROM) */
B93      dcl  prom_major_rel_version char (3) aligned;     /* emu major release version
    (offset 80 in ID PROM) */
B94      dcl  prom_minor_rel_version char (3) aligned;     /* emu minor release version
    (offset 83 in ID PROM) */
B95      dcl  prom_patch_version    char (3) aligned;      /* emu patch version (offset 86
    in ID PROM) */
B96      dcl  prom_iteration_number char (3) aligned;      /* emu iteration number (offset
    89 in ID PROM) */
B97      dcl  prom_build_number     char (8) aligned;      /* emu build number (offset 92
    in ID PROM) */
B98      dcl  prom_release_type     char (1) aligned;      /* emu release type (offset 100
    in ID PROM) */
B99      dcl  prom_version_text     char (29) aligned;     /* emu version text (offset 101
    in ID PROM) */
```

```
B100     dcl  prom_build_arch       char (20) aligned;      /* emu build architecture
    (offset 130 in ID PROM) */
B101     dcl  prom_build_os         char (20) aligned;      /* emu build OS (offset 150 in
    ID PROM) */
B102     dcl  prom_target_arch      char (20) aligned;      /* emu build architecture
    (offset 170 in ID PROM) */
B103     dcl  prom_target_os        char (20) aligned;      /* emu build OS (offset 190 in
    ID PROM) */
B104
B105     dcl  prom_build_number2    char (8) aligned;       /* possibly trimmed build
    number */
B106
B107     dcl  table_value fixed bin;


A570              then                                        /* if DPS8 cpu... */
Changed by B to:
B585              then do;                                    /* if DPS8 cpu... */


A579                        call syserr (4, "CPU ^a: Model #: ^a; Serial #: ^a; Ship date:
    ^a.",
A580                                                        /* log info from id prom */
A581                            substr ("ABCDEFGH", tag + 1, 1), cpu_model, cpu_serial,
    cpu_ship_date);
A582                        end;
Changed by B to:
B594
B595                        call privileged_mode_ut$read_id_prom (prom_layout_version, 60);
B596
B597                        /* log info from id prom */
B598
B599                        if prom_layout_version = "\377" then do;
B600                            if cpu_model = "DPS 8/SIM M" then
B601                               call syserr (BEEP, "Warning: You are running an old
    version of the DPS8M simulator. ^/^-Pleas
\ce update to R3.0.1 or later. See https://dps8m.gitlab.io/ for details.");
B602                            call syserr (ANNOUNCE, "CPU ^a: Model #: ^a; Serial #: ^a;
    Ship date: ^a.",
B603                               substr ("ABCDEFGH", tag + 1, 1), cpu_model, cpu_serial,
    cpu_ship_date);
B604                        end;
B605                        else do;
B606                            call privileged_mode_ut$read_id_prom (prom_commit_date, 70);
B607                            call privileged_mode_ut$read_id_prom
    (prom_major_rel_version, 80);
B608                            call privileged_mode_ut$read_id_prom
    (prom_minor_rel_version, 83);
B609                            call privileged_mode_ut$read_id_prom (prom_patch_version,
    86);
B610                            call privileged_mode_ut$read_id_prom
    (prom_iteration_number, 89);
```

```
B611                            call privileged_mode_ut$read_id_prom (prom_build_number,
    92);
B612                            call privileged_mode_ut$read_id_prom (prom_release_type,
    100);
B613                            call privileged_mode_ut$read_id_prom (prom_version_text,
    101);
B614                            call privileged_mode_ut$read_id_prom (prom_build_arch, 130);
B615                            call privileged_mode_ut$read_id_prom (prom_build_os, 150);
B616                            call privileged_mode_ut$read_id_prom (prom_target_arch,
    170);
B617                            call privileged_mode_ut$read_id_prom (prom_target_os, 190);
B618
B619                            prom_build_number2 = rtrim(prom_build_number, "\377");
B620                            if prom_build_number2 = "" | substr(prom_build_number,1,1)
    = " " then
B621                                prom_build_number2 = "<None>";
B622
B623                            call syserr (ANNOUNCE, "CPU ^a: Model #: ^a; Serial #: ^a;
    Ship date: ^a; PROM Layout Version: ^
\ca; ^/^-Simulator Release: ^a^a (^a); Build Number: ^a; ^/^-Build Arch: ^a; Build OS: ^a;
    ^/^-Target Arch: ^a; Target OS: ^a.",
B624                                substr (LETTERS, tag + 1, 1), cpu_model, cpu_serial,
    cpu_ship_date, prom_layout_version,
B625                                prom_release_type, rtrim(prom_version_text,"\377"),
    prom_commit_date,
B626                                prom_build_number2, prom_build_arch, prom_build_os,
    prom_target_arch, prom_target_os);
B627                        end;
B628                    end;
B629                end;


A586
Deleted by B, preceding:
B633                 do i = 1 to 8;

Comparison finished: 4 differences, 68 lines.
```

From the above, it can be seen that the entry `privileged_mode_ut$read_id_prom` is used to read the fields from the PROM. This is the same way that the original code read the three existing fields.

In future versions of Multics, if a need arises to check for a specific version of the simulator, the `prom_major_rel_version`, `prom_minor_rel_version`, `prom_patch_rel_version`, and `prom_iteration_number` fields can be checked, and if they represent a version less than that of one that fixes known problems, a version-specific warning message can be emitted. For now, the version number in the warning message uses R3.0.1, which is expected to be the first version of the simulator that correctly supports the new ID PROM fields.

3 Limitation of Proposed Fix

Because the L68 processor does not have a PROM, if a user reconfigures the Multics boot to make the booload processor a L68 processor, the `tc_init.pl1` fix above will not display any message on the operator console (nor log it in the syserr log). If the user configures other processors to be DPS 8/M processors, there will be PROMs in those processors and the system could have displayed a message when these secondary processors are added. The program that logs to the syserr log the CPU model and serial is `start_cpu`. One could add support there to note that the bootload processor was a L68 and therefore didn't log the simulator version information, and check if a newly-added CPU was a DPS 8/M processor, and if so, log the same console/syserr message as is done by `tc_init.pl1`. However, the logic to do this could get complicated because ideally, the message should only be emitted one time, so subsequent CPU starting should not log the message if one has already been logged. Also, `start_cpu.pl1` would have to examine the bootload processor to determine its CPU type and perform other checks to make sure that it had a PROM and therefore caused a message to be emitted already. Alternately, state could be kept to indicate if the message had been logged already and then this state checked by each invocation of `start_cpu`.

Due to the complexities in being thorough in this case, and due to the fact that only developers who need to exercise L68-only features or run ISOLTS tests on a system that was booted from a L68 CPU will probably ever change the CPU configuration to include L68 CPUs, the added complexity (and additional hardcore code necessary to implement this logic) is not worth the effort.

4 Documentation

The simulator documentation already documents the currently supported PROM contents, and the simulator "show prom" command can be used to display the layout, complete with the fields that are supported by that version of the simulator. This is probably sufficient.

At this time, no Multics errata info segment documentation changes are proposed. However, we may wish to create such an errata for the "Multics Differences Manual DPS 8 70/M", even though this wasn't an official Multics manual.

5 Test

The changes will be tested by creating a new Multics System Tape (MST) with the updated code, and running it on old and newer versions of the simulator, with the bootload processor specified as both a DPS 8/M processor and a L68 processor.

6 Version History

2023-07-05	1.0	Initial version of the MCR.
2023-07-07	1.1	Updated based on MCR review comments.
2023-07-09	1.2	Correct typos and add Build Number PROM field.
2023-07-09	1.3	Added note about the limitations of the proposed fix.
2023-07-11	1.4	Updates to how build number is displayed.
2023-07-15	1.5	Updated cpa output and clarified there are two potential syserr/console messages.