MCR10119
Fix Unexpected Fault During BCE Reinit Due to Race Condition
Revision 1.0

Eric Swenson / Charles Anthony

August 22, 2022

1 Problem Description

Occasionally, after recent changes to the dps8 simulator, Multics will get an Unexpected Fault during the BCE `reinit` request. This occurs due to a race condition where the Timer Runout fault vectors point to the Unexpected Fault handlers for a brief period of time after a `reinit`. If the config deck is edited at BCE, and within 1/8 of a second after executing the `w` request to write the config deck, the `reinit` request is issued, it is possible to hit the race condition. Of course, it is nearly impossible to do this manually, as human response time is too slow. However, using scripts to perform a reconfiguration and boot, it is quite possible to hit the race condition.

The ticket for this issue is https://multics-trac.swenson.org/ticket/287

2 Analysis

While testing some fixes to the simulator and specifically running ISOLTS, Charles Anthony discovered an intermittent "Unexpected Fault" occurring during the BCE `reinit` request. The scripts he was running would make changes to the config deck, and then, in order to make BCE process the changes, would issue the `reinit` request. In previous versions of the simulator, this would succeed, but he started seeing occasional faults during the execution of this request.

A `dump -long` copied over to another Multics system and analyzed was not helpful. It appeared that the stack history of the unexpected fault was obscured by BCE's handling of the fault and subsequent return to `bce (early)` command level.

Charles created a script that would loop 1000 times, making changes to the config deck and then issuing the `reinit` command. He was able to see the failure after about 30-50 tries, proving that the issue is truly intermittent. Unfortunately, rebuilding the simulator with the TESTING flag enabled make it so the issue didn't occur, so use of the TESTING simulator could not be used to find the unexpected fault.

After adding some `printf` debugging to the simulator, Charles was able to determine that the unexpected fault was a Timer Run Out (TRO) fault. The fault was occurring during the execution

of `initialize_faults`. Charles instrumented the Load Timer Register (LDT) instruction to show where it was being loaded and with what value as the operand. The TImer Register is loaded with -1 (about 4 minutes) early in the boot process, but BCE was repeatedly setting it to 65536 (about 1/8 of a second) and letting it timeout. The Timer Register was determined to be used in various page event completion routines to detect paging failures.

The BCE `reinit` request command is very simple – it simply sets the boot phase back to prior to the config deck parsing, causing `initialize_faults` to be run. This procedure is divided into two parts: the first part sets most fault and interrupt vectors to raise "unexpected fault" conditions; the second part sets up the fault vectors for BCE. This includes redirecting the TRO fault to the `pxss` timeout handler. For a very brief period of time after a `reinit` request, the TRO fault vector will point to the Unexpected Fault handler.

The following sequence of events will cause an Unexpected Fault crash of BCE:

- A paging event occurs (in this case, this will be the `w` config deck editor request having written to the page), which causes the Timer Register (TR) to 1/8 second. When it runs out, it will get reset so that it will run out every 4 minutes thereafter.
- The `reinit` BCE request is issued.
- `initializer_faults` runs and opens the race window.
- If the 1/8 second TRO occurs here, an Unexpected Fault will occur.
- Otherwise, the race window closes.

3 Proposed Fix

The proposed fix is to reset the Timer Register (TR) to its maximum value (-1) as part of the `reinit` command. This will prevent a TRO from occurring in the brief period of time that the race window is open. At the next paging event, the TR will be set back to 1/8 second, but by this time, the race window will have closed.

The program `bce_get_to_command_level.pl1` will be changed as follows:

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

Inserted in B:
B60      dcl  privileged_mode_ut$ldt ext entry (fixed bin);     /* to load timer register */
Preceding:
A60      dcl  request_abort_                condition;


Inserted in B:
B132             call privileged_mode_ut$ldt (-1);              /* Reset timer in case
    pxss has started it; */
B133                                                            /* prevents unexpected
    fault during initialize_faults. */
Preceding:
A131             go to Boot_label;

Comparison finished: 2 differences, 3 lines.
```

4 Testing

A Multics tape was created with the above fix. Charles used this tape to prove that he could not reproduce the unexpected fault in a script that iterated 1000 times updating the config deck and issuing the `reinit` command. Prior to the fix, the unexpected fault would occur after about 30-50 iterations.

In addition, the GHM_Test system has been running with this fix as well with no issues.

Our CI/CD process for simulator changes runs a series of tests which begin with cold booting a Multics system, configuring a system, and then running a bunch of texts. It culminates with running ISOLTS. This process will be run (has been run once already) using a hardcore with the above fix.

5 Documentation

There are no documentation changes required by this change apart from comments in the code and history comments.

6 Version History

2022-08-22	1.0	Initial version of the MCR.