z/OS V1R13

z/OS UNIX DBX: Hookless debug support

Overview

- Problem Statement / Need Addressed
  - When code is compiled for debug it results in a program that is larger, runs slower, and sometimes does allow problems seen in production code to be reproduced.
- Solution
  - dbx can debug a program compiled with the -qdebug=nohook or -Wc,"DEBUG(NOHOOK).
- Benefit / Value
  - Generated code is closer to production code
  - Generated code runs faster
  - Generated code is smaller

Usage and invocation

```
/u/clbates > xlc  -g  -qdebug=nohook  test.c

/u/clbates > cc  -g   -Wc,"DEBUG(NOHOOK)"   test.c

/u/clbates > dbx a.out

(dbx64) s

stopped in main at line 13 in file "test.c"  ($t1)

   13       foo();

(dbx64)
```

Notice you don't have to do a *stop in main* then continue, you can just step!

dbx can only stop on lines for which there is code generated. Notice that stepping into foo does not stop on the first line of foo but on the first statement in foo:

```
stopped in main at line 13 in file "test.c"  ($t1)

   13       foo();

(dbx64)  s

stopped in foo at line 5 in file "test.c"  ($t1)

    5       i = i+3;

(dbx64)
```

Look No EX hooks

```
(dbx64) listi

0x22393974 (foo+0x4c)  5800d098     L       R0,152(,R13)

0x22393978 (foo+0x50)  a70a0003     AHI     R0,3

0x2239397c (foo+0x54)  5000d098     ST      R0,152(,R13)

0x22393980 (foo+0x58)  5800d098     L       R0,152(,R13)

0x22393984 (foo+0x5c)  a70a0001     AHI     R0,1

0x22393988 (foo+0x60)  5000d098     ST      R0,152(,R13)

0x2239398c (foo+0x64)  5800d098     L       R0,152(,R13)

0x22393990 (foo+0x68)  a70a000a     AHI     R0,10

0x22393994 (foo+0x6c)  5000d098     ST      R0,152(,R13)
```

If you do not compile with the -qdebug=nohook or -Wc,"DEBUG(NOHOOK)

```
(dbx64) listi

0x22393978 (foo+0x50)  4400c1ac     EX      0,428(,R12)

0x2239397c (foo+0x54)  5800d098     L       R0,152(,R13)

0x22393980 (foo+0x58)  a70a0003     AHI     R0,3

0x22393984 (foo+0x5c)  5000d098     ST      R0,152(,R13)

0x22393988 (foo+0x60)  4400c1ac     EX      0,428(,R12)

0x2239398c (foo+0x64)  5800d098     L       R0,152(,R13)

0x22393990 (foo+0x68)  a70a0001     AHI     R0,1
```

```
0x22393994 (foo+0x6c)  5000d098     ST      R0,152(,R13)
0x22393998 (foo+0x70)  4400c1ac     EX      0,428(,R12)
0x2239399c (foo+0x74)  5800d098     L       R0,152(,R13)
```

- Debug programs whether they are compiled with EX hooks or not.
- Think of an EX hook as a compiled in breakpoint.
  - When dbx turns them on it stops at everyone of them.
  - With this support we never turn them on. They are like noops to dbx.
- Dbx may behave different if the program is compiled with hooks.
- Consider

```
if (TRUE)
        I = I + 5;
```

- There may not be any code generated for the **if (TRUE)** so when stepping, you may step over it if you compiled without hooks.
- If you compiled with hooks there would be a lone EX hook associated with the **if (TRUE)** statement so dbx would be able to set a breakpoint there and step through both lines.
- dbx will step one line at a time by default.
- dbx -DS a.out will make dbx to step one statement at a time.

Consider:

```
while ( s[i] != 0 ) i++;
```

Normally you can step over this line with a single step but if -DS is specified step will stop each time through the loop.

Interactions and dependencies

- Software Dependencies
  - None
- Hardware Dependencies
  - None
- Exploiters
  - Dbx requires z/OS V1R13 Common Debug Architecture runtime.

Migration and coexistence considerations

- None

Installation

- No Changes

Session summary

- Debug of code compiled without exhooks:
  - Machine code closer to production code
  - Smaller debuggable programs
  - Faster debuggable programs
  - Step can now be used to step into the first line of the program

Appendix - References

- SA22-7802 z/OS UNIX System Services Command Reference
- SA22-7807 z/OS UNIX System Services Messages and Codes
- SA22-7805 z/OS UNIX System Services Programming Tools
- SA22-7801 z/OS UNIX System Services User's Guilde