RENESAS Tool News

RENESAS TOOL NEWS on August 1, 2004: RSO-M3T-PD32RM_1-040801D

A Note on Using Emulator Debuggers M3T-PD32RM, M3T-PD32R, M3T-PD308F, M3T-PD308, M3T-PD30F, and M3T-PD30

Please take note of the following problem in using the M3T-PD32RM, M3T-PD32R, M3T-PD308F, M3T-PD308, M3T-PD30F, and M3T-PD30 emulator debuggers:

- On using the C Watch window

1. Products and Versions Concerned

M3T-PD32RM V.2.00 Release 1 and V.2.10 Release 1
used with the M32100T5-SDI-E, M32100T3-SDI-E, and M32100T-EZ-E emulators for the M32R family MCUs
M3T-PD32R V.4.00 Release 1 and V.4.10 Release 1
used with the M32100T2-SDI-E emulators for the M32R family MCUs
M3T-PD308F V.3.00 Release 1 through V.3.20 Release 1
for the M32C/80 series MCUs
M3T-PD308 V.5.00 Release 1 through V.5.20 Release 1
for the M32C/80 and M16C/80 series MCUs
M3T-PD30F V.2.00 Release 1 through V.2.20 Release 1
for the M16C/60 and /Tiny series MCUs
M3T-PD30 V.8.00 Release 1 through V.8.20 Release 1
for the M16C/60, /30, /20, and /10 series MCUs

2. Description

If structures, unions, classes, arrays, and pointers are expanded, the emulator debugger may hang up.

2.1 Conditions

This problem occurs if structures, unions, classes, arrays, and pointers (hereafter called variables) are concerned with any of the following conditions:

(1) Any variable except an array that has already been declared to be typedef is again declared so.

In Example 1 shown below, the emulator debugger hangs up when s2 and p2 are expanded.

Note, however, that when s1 and p1 and pS1 (a pointer-type variable declared to be typedef) and s3 (a structure-type variable declared to be typedef) are expanded, no problems arise.

Example 1:

```
--------------------------------------------------------------
typedef struct tagS1 {
   int a;

   int b;
} STRUCT1;
STRUCT1 s1;                    /* Not concerned */

typedef unsigned char* PSTR1;
PSTR1 p1;                     /* Not concerned */

typedef STRUCT1 STRUCT2;
STRUCT2 s2;                    /* Concerned */

typedef PSTR1 PSTR2;
PSTR2 p2;                     /* Concerned */

typedef struct tagS3 {
   STRUCT1 s;
   int c;
} STRUCT3;
STRUCT3 s3;                    /* Not concerned */

typedef STRUCT1* PSTRUCT1;
PSTRUCT1 pS1;                   /* Not concerned */
--------------------------------------------------------------
```

Conditions (2) and later are applied only to the object files to debug being written in the ELF/DWARF2 format; otherwise not concerned.

(2) Any variable except an array that has already been declared to be typedef is qualified as const or volatile.

Example 2:

```
------------------------------------------------------
volatile STRUCT1 vs1;              /* Concerned */
const PSTR1 cp1;                   /* Concerned */
------------------------------------------------------
```

(3) Any variable except an array that has already been qualified as const or volatile is declared to be typedef with another name. In Example 3 below, the expansion of cpstr is not concerned since cpstr is not of type pointer qualified as const or volatile.

Example 3:

```
------------------------------------------------------
typedef unsigned char* const PCSTR;
PCSTR pcstr;                     /* Concerned */
typedef const unsigned char* CPSTR;
CPSTR cpstr;                     /* Not concerned */
------------------------------------------------------
```

(4) Any variable is qualified as const-volatile (in this condition, an array is included). In Example 4 below, the expansion of vpcstr and that of cpvstr are not concerned since vpcstr and cpvstr are not of type pointer qualified as const or volatile.

Example 4:

```
------------------------------------------------------
unsigned char* const volatile pcvstr; /* Concerned */
volatile unsigned char* const vpcstr; /* Not concerned */
const unsigned char* volatile cpvstr; /* Not concerned */
------------------------------------------------------
```

3. Workaround

This problem can be circumvented in either of the following ways:

(1) Reference any variable concerned using the print command in the Script window, if necessary.

(2) In conditions (1) and (3), use #define instead of typedef.

Example

```
------------------------------------------------------
#define STRUCT2 STRUCT1
STRUCT2 s2;
------------------------------------------------------
```

4. Schedule of Fixing the Problem

We plan to fix this problem in our next release of the products.

[Disclaimer]

The past news contents have been based on information at the time of publication. Now changed or invalid information may be included. The URLs in the Tool News also may be subject to change or become invalid without prior notice.