Microcontroller Technical Information

RA78K0S 78K0S Assembler Package Usage Restrictions	Document No.	ZBG-CD-07-0048	1/2
	Date issued	August 30, 2007	
	Issued by	Development Tool Solution Group Multipurpose Microcomputer Systems Division Microcomputer Operations Unit NEC Electronics Corporation	
Related documents RA78K0S Ver. 1.40 or Later - Operation: U16656EJ1 (1st) RA78K0S Ver. 1.30 or Later - Language: U14877EJ1 (1st) RA78K0S Structured Assembly Language: ST78K0S V1.00 or Later U11623EJ1 (1st) PM plus Ver. 5.10: U16569EJ1 (1st) 78K0S Series Assembler Package RA78K0S V1.40 Operating Precautions: SUD-DT-03-0408-E	Notification classification	√	Usage restriction
			Upgrade
			Document modification
			Other notification

1. Affected products

   RA78K0S V1.31/V1.40

2. Restrictions

   The following restrictions (No. 20 to No. 23) have been added. See the attachment for details.

   - No. 20 An A106 error is output if -Dsymbol option is specified in the RA78K0S.
   - No. 21 An A105 error is output in the list converter if the definition of a function name that has been referred to externally from a C source is a PUBLIC symbol in a structured assembler source or an assembler source.
   - No. 22 An A106 error is output in the list converter if coding of the segment name is omitted in the ORG quasi directive.
   - No. 23 Only the first one item in an absolute assembler list is output if multiple C sources are registered and the assembler source output option is specified.

3. Workarounds

   The following workarounds are available for these restrictions. See the attachment for details.

   - No. 20 Do not specify the “symbol definition” option (assembler option) when specifying the assembler source module output option in the CC78K0S.
   - No. 21 Do not start the list converter.
   - No. 22 Describe the segment name for the ORG quasi directive.
   - No. 23 Create an absolute assembler list by using the LC78K0S started via the command prompt.

4. Modification schedule

Restrictions No. 20 to No. 23 will be corrected in RA78K0S V2.00, which is planned for release in September 2007.

* For the detailed release schedule of modified products, contact an NEC Electronics sales representative.

5. List of restrictions

A list of restrictions in the RA78K0S, including the revision history and detailed information, is described on the attachment.

6. Document revision history

78K0S Assembler Package RA78K0S - Usage Restrictions

Document Number	Date Issued	Description
SBG-DT-04-0006	January 23, 2004	Newly created.
SBG-DT-04-0113	March 12, 2004	Addition of restrictions (No. 17 to No. 19)
ZBG-CD-07-0048	August 30, 2007	Addition of new restrictions (No. 20 to No. 23)

List of Restrictions in RA78K0S

1. Product History

No.	Bugs and Changes/Additions to Specifications	Version		
		V1.30	V1.31	V1.40
1	The assembler does not operate correctly when "^Z" is not placed at the end of an include file. (Windows version only)	○	○	○
2	An error will be output from the command prompt if there is no device file when the structured assembler is started from the command prompt.	○	○	○
3	Even if a BR quasi directive for which no branch destination is specified is described in the assembler source, an error is output not to the assembler but to the linker.	×	○	○
4	A multi-byte character comment described in a macro definition in an assembler source is not displayed in the macro expansion in the assemble list.	×	○	○
5	Even if a numeric value is specified for a symbol using the -D option, the macro can use it only as a truth value.	×	○	○
6	An invalid code may be output as a result of calculating the number of codes after the BR directive in the last segment of the assembler source is optimized.	×	○	○
7	If a branch quasi directive that has not been resolved in the syntax analysis path exists in the absolute code segment in an assembler source, and referencing with a HIGH/LOW operator is performed for a backward reference symbol, an invalid code is output.	×	○	○
8	A segment for which “MERGE AT” is specified by a directive is not prioritized for allocation.	×	○	○
9	The debugger or simulator outputs an error if the ancillary symbol table output from the compiler exists in the PUBLIC/EXTRN symbol.	×	○	○
10	When the segment name consists of eight characters, the linker may output an invalid segment name in the error message.	×	○	○
11	An error occurs if a control statement is crossed in a structured assembly language description.	×	×	×
12	The assembler performs illegal processing if the label receiving the effect of optimization is described in the *saddr* part when an EQU definition is performed for a bit symbol with the value *saddr.bit*.	×	×	×
13	Concatenate (&) will not be linked if the macro quasi directive IRP is nested.	×	×	×
14	The dependence of include files when the structured assembler is started by the Project Manager.	×	×	×
15	The dependence of include files when the assembler is started by the Project Manager.	×	×	×
16	An error occurs when the project file created by ID78K0S-NS (V2.xx) or SM78K0S (V2.xx) operating alone is opened by the Project Manager.	×	×	○
17	Allocation of segments whose size is 0 may be invalid.	×	×	×
18	Name definition in the assembler results in an A106 error in the object converter.	×	×	×
19	An A402 error is output in the linker if the source file name consists of 15 or more characters.	×	×	×

×: Applicable, ○: Not applicable, –: Not relevant

No.	Bugs and Changes/Additions to Specifications	Version		
		V1.30	V1.31	V1.40
20	An A106 error is output if -Dsymbol option is specified in the RA78K0S.	×	×	×
21	An A105 error is output in the list converter if the definition of a function name that has been referred to externally from a C source is a PUBLIC symbol in a structured assembler source or an assembler source.	×	×	×
22	An A106 error is output in the list converter if coding of the segment name is omitted in the ORG quasi directive.	×	×	×
23	Only the first one item in an absolute assembler list is output if multiple C sources are registered and the assembler source output option is specified.	×	×	×

×: Applicable, ○: Not applicable, –: Not relevant

2. Details of Restrictions

No. 1 The assembler does not operate correctly when "^Z" is not placed at the end of an include file. (Windows version only)

[Description]

The absence of "^Z" at the end of an include file may cause an error or the output of invalid debug information.

[Workaround]

Enter a line containing only ^Z at the end of the include file.

[Correction]

This issue has been corrected in V1.30.

No. 2 An error will be output from the command prompt if there is no device file when the structured assembler is started from the command prompt.

[Description]

An error will be output from the command prompt or no response will be sent from the program if there is no device file when the structured assembler is started from the command prompt.

[Workaround]

Make sure that the device file is installed. In addition, when the folder in which the structured assembler or device file is installed is not specified by default, specify the path for the device file using the -Y option.

[Correction]

This issue has been corrected in V1.30.

No. 3 Even if a BR quasi directive for which no branch destination is specified is described in the assembler source, an error is output not to the assembler but to the linker.

[Description]

If a BR quasi directive for which no branch destination is specified is described in the assembler source, no error is output in the assembler but the error “F304 Operand out of range (segment ‘segment name’, address xxxxH, type addressing type’” is output in the linker.

[Workaround]

When the project manager is used, check the item “Output Cross Reference List” on the [Output2] tab in the Assembler Options dialog box.

When the RA78K0S is started from the command line, add the -kx option to the RA78K0S.

After build is executed and linking is complete, a cross reference list is created in the latter part of the list file (*.prn).

Asterisks (******* * ***** ****) are displayed in this list to indicate undefined symbols.

Example:

If the cross reference list is output as shown below, this means BBBBBBBB is an undefined symbol.

```
AAAAAAAA  FE44H ADDR  SHORT  174#  873  1601
BBBBBBBB  ******* * ***** ****    1250
CCCCCCCC  306H  ADDR  PROG1 1008  1032  1046  1303#
```

[Correction]

This issue has been corrected in V1.31.

No. 4 A multi-byte character comment described in a macro definition in an assembler source is not displayed in the macro expansion in the assemble list.

[Description]

If a multi-byte character comment is described in a macro definition, only the multi-byte character comment cannot be displayed in macro expansion in the assemble list file created after assembly.

Example:

sample.asm

```
NAME sample
MAC1 MACRO
        NOP     ; comment1 Multi-byte character  Single-byte Japanese syllabary
        ENDM

CSEG
SAMPLE1:
        MAC1

        END
```

sample.prn

Assemble list

```
ALNO  STNO  ADRS  OBJECT  M I  SOURCE STATEMENT

1     1                          NAME    SAMPLE
2     2                    M     MAC1    MACRO
3     3                    M             NOP     ; comment1 Multi-byte character
                                                   Single-byte Japanese syllabary
4     4                    M             ENDM
5     5
6     6     ----                         CSEG
7     7     0000                 SAMPLE1:
8     8                                  MAC1
      9     0000  00       #1            NOP     ; comment1  Single-byte
                                                   Japanese syllabary
9     10
10    11                                 END
```

[Workaround]

There is no workaround.

[Correction]

This issue has been corrected in V1.31.

No. 5 Even if a numeric value is specified for a symbol using the -D option, the macro can use it only as a truth value.

[Description]

A symbol value specified using the -D option in the assembler source instead of describing $set/$reset/EQU does not become valid.

Example 1: (2) may be valid even if “other than -Dsyma=0” is specified

Normally, the A109 error should be output if there is a $set(syma)/$reset(syma) description when -D is specified, but $set(syma)/$reset(syma) becomes valid and the -D option is ignored.

```
$if (syma)
  ; (1) When “-Dsyma” or “other than -Dsyma=0” is specified
$else
  ; (2) When “-Dsyma=0” is specified
$endif
```

Example 2: The -D option is ignored and F303 or F407 will be output.

Normally, the A109 error should be output if there is an EQU description when -D is specified, but EQU becomes valid because -D is ignored.

```
$_if ( symb = 0FFH )
  ; (3) When “-Dsymb=0FFH” is specified
$_elseif ( symb = 01H )
  ; (4) When “-Dsymb” or “-Dsymb=1” is specified
$else
  ; (5) When a value other than (3) or (4) is specified for symb
$endif
```

[Workaround]

Describe $set(syma)/$reset(syma)/EQU without using the -D option.

[Correction]

This issue has been corrected in V1.31.

No. 6 An invalid code may be output as a result of calculating the number of codes after the BR directive in the last segment of the assembler source is optimized.

[Description]

An invalid code may be output as a result of calculating the number of codes after the BR directive in the last segment of the assembler source is optimized.

Though information in the assemble list file (*.prn) and link list file (*.map) is correct, information in the relocatable object file (*.rel) is invalid. Consequently, information in the load module file (*.lmf) and HEX-format object module file (*.hex) also becomes invalid.

[Workaround]

Implement any of the following workarounds.

- Use the BR instruction instead of the BR quasi directive because its instruction length is clear.
- Insert NOPs for the amount of insufficient code at the end of the segment in which codes are lacking.

[Correction]

This issue has been corrected in V1.31.

No. 7 If a branch quasi directive that has not been resolved in the syntax analysis path exists in the absolute code segment in an assembler source, and referencing with a HIGH/LOW operator is performed for a backward reference symbol, an invalid code is output.

[Description]

If a branch quasi directive that has not been resolved in the syntax analysis path exists in the absolute code segment in an assembler source, and referencing with a HIGH/LOW operator is performed for a backward reference symbol, an invalid code is output.

[Workaround]

Describe the BR instruction which has a clearly defined instruction length instead of the BR quasi directive.

[Correction]

This issue has been corrected in V1.31.

No. 8 A segment for which “MERGE AT” is specified by a directive is not prioritized for allocation.

[Description]

A segment for which “MERGE AT” is specified by a directive must be prioritized for allocation to the address specified by AT, but is not. Consequently, a segment without AT specification is allocated first, which may cause the error F206 or F304.

[Workaround]

Move the already allocated segment to the area where the segment for which “MERGE AT” is specified is to be allocated.

[Correction]

This issue has been corrected in V1.31.

No. 9 The debugger or simulator outputs an error if the ancillary symbol table name output from the compiler exists in the operand field of the PUBLIC/EXTRN quasi directive.

[Description]

Because of a linker bug, the debugger or simulator outputs the error message “b019(A): Cannot seek file” if there is an ancillary symbol table name output from the in the operand field of the PUBLIC/EXTRN quasi directive.

[Workaround]

There is no workaround.

[Correction]

This issue has been corrected in V1.31.

No. 10 When the segment name consists of eight characters, the linker may output an invalid segment name in the error message.

[Description]

When the segment name consists of eight characters, the linker may output an invalid segment name in the error message.

[Workaround]

There is no workaround.

[Correction]

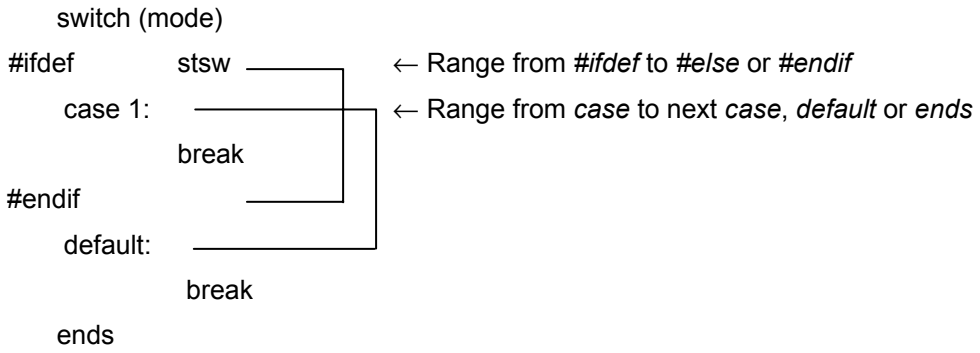
This issue has been corrected in V1.31.

No. 11 An error occurs if a control statement is crossed in a structured assembly language description.

[Description]

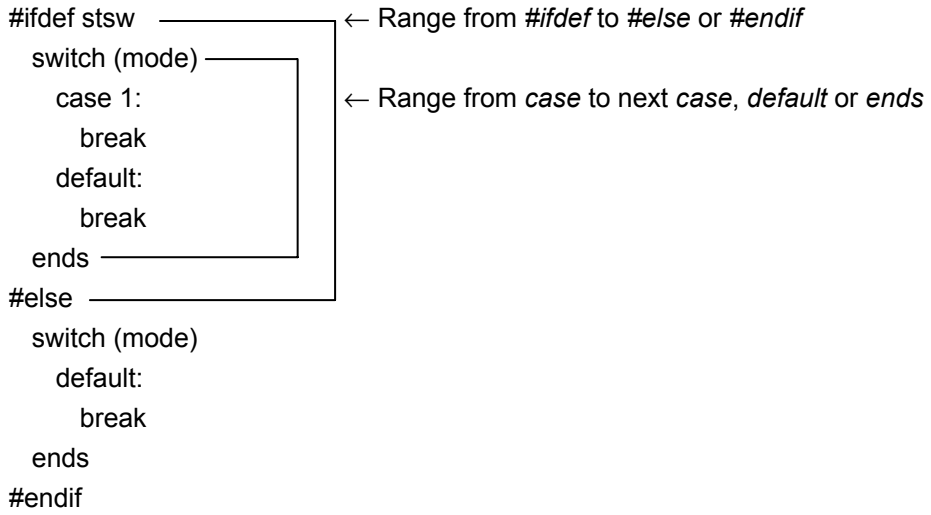
If a control statement is divided or crossed by *#ifdef* to *#endif*, an error occurs when *#ifdef* is true.

Example:



[Workaround]

There is no error in the case of nesting. The source should be rewritten so that the ranges of the control statements do not cross.



[Correction]

Regard this issue as a usage restriction.

No. 12 The assembler performs illegal processing if the label receiving the effect of optimization is described in the *saddr* part when an EQU definition is performed for a bit symbol with the value *saddr.bit*.

[Description]

The assembler performs illegal processing if the label receiving the effect of optimization is described in the *saddr* part when an EQU definition is performed for a bit symbol with the value *saddr.bit*.

[Phenomenon]

The assembler performs illegal processing in the following cases.

- When *saddr.bit* is 0FD20H, path 1 of a label is outside the area, and path 2 is inside the area, an error is output in path 1 for the EQU definition line, but not output in path 2. At this time, the object is created but it is incorrect.
- When *saddr* is 0FF1FH, path 1 of a label is inside the area, and path 2 is outside the area, no error is output in path 1 for the EQU definition line, while an error is output in path 2. The assembly error “F410 Phase error” will be output for a label that is defined after this EQU symbol has been referred to.

  When this label is referred to, the object becomes incorrect.

[Workaround]

There is no workaround.

[Correction]

Regard this issue as a usage restriction.

No. 13 Concatenate (&) will not be linked if the macro quasi directive IRP is nested.

[Description]

Nesting IRP that includes the character string concatenation symbol "&” will render the macro expansion results invalid since the parameter is not converted.

Example:

```
IRP     ZZZ, <1,2,3>
        IRP   XXX, <4,5,6>
              LABEL&ZZZ&XXX:     ; Incorrect
        ENDM
ENDM
```

[Workaround]

Do not use “&” when an IRP is nested.

[Correction]

Regard this issue as a usage restriction.

No. 14 The dependence of include files when the structured assembler is started by the Project Manager.

[Description]

In checking the dependence of include files described in structured assembler language when creating a make file in the Project Manager, only the deletion of comments and character strings is processed, and conditions such as *#ifdef* are ignored.

[Phenomenon]

In the description example below, the *#ifdef*, *#else*, and *#endif* lines are ignored. As a result, the existence of the file name specified by *#include* is checked regardless of whether the file is referred to. An error will occur if the file specified by *#include* does not exist when executing a build.

Example:

```
#ifdef SYM
#include "func1.inc"
#else
#include "func2.inc"
#endif
```

[Workaround]

There is no workaround.

[Correction]

Regard this issue as a usage restriction.

No. 15 The dependence of include files when the assembler is started by the Project Manager.

[Description]

In checking the dependence of include files described in assembler language when creating a make file in the Project Manager, deletion of comments and character string processing are performed but conditions such as *$if* and *$_if* are ignored.

[Phenomenon]

Lines *#ifdef*, *#else* and *#endif* are ignored in the following example. As a result, the existence of the file name specified by *#include* is checked regardless of whether the file is referred to. An error will occur if the file specified by *#include* does not exist when executing a build.

Example:

```
#if (SYM)
#include(func1.h)
#else
#include(func2.h)
#endif
```

[Workaround]

There is no workaround.

[Correction]

Regard this issue as a usage restriction.

No. 16 An error occurs when the project file created by ID78K0S-NS (V2.xx) or SM78K0S (V2.xx) operating alone is opened by the Project Manager.

[Description]

The error message “F150: Failed to read the project file.” is output if an attempt is made to read a project file created by ID78K0S-NS (V2.xx) or SM78K0S (V2.xx), which is started solely (a file that is created by tools other than the Project Manager).

[Workaround]

Create a new project file in the Project Manager that has the same name as the project file created by ID78K0S-NS (V2.xx) or SM78K0S (V2.xx), which is started solely. After that, select [Select Debugger] from the [Option] menu in the Project Manager and specify the debugger (ID or SM) used.

[Correction]

This issue has been corrected in V1.40.

No. 17 Allocation of segments whose size is 0 may be invalid.

[Description]

Allocation of segments may be invalid when the following three conditions are satisfied.

(1) A segment whose size is 0 exists.
(2) The segment of (1) is not addressed.
(3) A segment that is addressed to a segment other than (1) exists.

A segment whose address is not output to the link list in the ascending address order is an illegally-allocated segment.

[Workaround]

Address the illegally allocated segment using the directive file.

[Correction]

This issue will be corrected in V2.00.

No. 18 Name definition in the assembler results in an A106 error in the object converter.

[Description]

Symbol information may be invalid when all the conditions shown below are satisfied, which results in an A106 error in the object converter. In addition, the error message “Ab019 : Reading of file went wrong.” will be output when the load module file is downloaded by the debugger.

(1) The name is defined in the structured assembler source or assembler source
(2) The name of (1) is externally referred to as a function name by the C source.
(3) The file of (1) is linked after the C source of (2).

[Workaround]

Link the C source to a location below where the structured assembler source or assembler source is linked.

[Correction]

This issue will be corrected in V2.00.

No. 19 An A402 error is output in the linker if the source file name consists of 15 or more characters.

[Description]

An A402 error is output in the linker when the following two conditions are satisfied.

(1) The source file name consists of 15 or more characters.
(2) Debug information is not output by the assembler.
The output of debug information is not specified in any of the following cases.
- “Release Build” is selected in PM or PM+.
- The “-nga -ng” option is specified in the assembler.

[Workaround]

Implement any of the following workarounds.

(1) Change the name of the source file so that it includes less than 15 characters.

(2) Select “Debug Build” in PM or PM+, or specify the “-ga -g” option (specified by default) in the assembler.

Debug information output by the assembler does not affect the generated HEX file.

[Correction]

This issue will be corrected in V2.00.

No. 20 An A106 error is output if -Dsymbol option is specified in the RA78K0S.

[Description]

If a *symbol* in the –Dsymbol option is specified in the RA78K0S for an assembler source file in which debug information is appended via the CC78K0S, invalid symbol information (symbol name that cannot be referred to) is output and the OC78K0S outputs an A106 error for the symbol table.

[Workaround]

Do not specify the “symbol definition” option (assembler option) when specifying the assembler source module output option in the CC78K0S. Specify the “symbol definition” option for an assembler source file by using the [Special Compiler Options...] menu.

[Correction]

This issue will be corrected in V2.00.

No. 21 An A105 error is output in the list converter if the definition of a function name that has been referred to externally from a C source is a PUBLIC symbol in a structured assembler source or an assembler source.

[Description]

An A105 error is output in the list converter if the definition of a function name that has been referred to externally from a C source is a PUBLIC symbol in a structured assembler source or an assembler source.

[Workaround]

Do not start the list converter.

[Correction]

This issue will be corrected in V2.00.

No. 22 An A106 error is output in the list converter if coding of the segment name is omitted in the ORG quasi directive.

[Description]

An A106 error is output in the list converter if coding of the segment name is omitted in the ORG quasi directive.

Example:

```
---------- Assembler source ----------
                ORG     1000H
                MOV             A, SADR2
                MOVW            AX, SADR3
```

[Workaround]

Describe the segment name for the ORG quasi directive.

Example:

```
---------- Assembler source ----------
        MAIN1   ORG     1000H
                MOV             A, SADR2
                MOVW            AX, SADR3
```

[Correction]

This issue will be corrected in V2.00.

No. 23 Only the first one item in an absolute assembler list is output if multiple C sources are registered and the assembler source output option is specified.

[Description]

Only the first one item in an absolute assembler list is output if multiple C sources are registered and the assembler source output option is specified.

[Workaround]

Create an absolute assembler list by using the LC78K0S started via the command prompt.

[Correction]

This issue will be corrected in V2.00.

3. Cautions

None.