FUJITSU SEMICONDUCTOR
CONTROLLER MANUAL

F2MC16LX-E

F²MC-16LX family
Standby Mode Transition Instruction Check Tool Manual

FUJITSU LIMITED

PREFACE

Objectives and Intended Readership

This document explains how to operate the tool for detected the standby transition instruction and to check whether it corresponds to the Standby Cancel Fail, in the customer programs developed for use with the F²MC-16LX.
The checking tool explained herein operates with a command line on the following OS; Windows 98/Windows Me/Windows NT4.0/Windows 2000/Windows XP.

Trademarks

- The contents of this document are subject to change without notice. Customers are advised to consult with FUJITSU sales representatives before ordering.
- The information and circuit diagrams in this document are presented as examples of semiconductor device applications, and are not intended to be incorporated in devices for actual use. Also, FUJITSU is unable to assume responsibility for infringement of any patent rights or other rights of third parties arising from the use of this information or circuit diagrams.
- The products described in this document are designed, developed and manufactured as contemplated for general use, including without limitation, ordinary industrial use, general office use, personal use, and household use, but are not designed, developed and manufactured as contemplated (1) for use accompanying fatal risks or dangers that, unless extremely high safety is secured, could have a serious effect to the public, and could lead directly to death, personal injury, severe physical damage or other loss (i.e., nuclear reaction control in nuclear facility, aircraft flight control, air traffic control, mass transport control, medical life support system, missile launch control in weapon system), or (2) for use requiring extremely high reliability (i.e., submersible repeater and artificial satellite).
  Please note that Fujitsu will not be liable against you and/or any third party for any claims or damages arising in connection with above-mentioned uses of the products.
- Any semiconductor devices have an inherent chance of failure. You must protect against injury, damage or loss from such failures by incorporating safety design measures into your facility and equipment such as redundancy, fire protection, and prevention of over-current levels and other abnormal operating conditions.
- If any products described in this document represent goods or technologies subject to certain restrictions on export under the Foreign Exchange and Foreign Trade Law of Japan, the prior authorization by Japanese government will be required for export of those products from Japan.

CONTENTS

1 General Description of F²MC-16LX Standby Mode Transition Instruction Check Tool

This section gives a general description of the F²MC-16LX Standby Mode Transition Instruction Check Tool.

General Description of Check Tool

This check tool detects standby mode transition instructions in the program developed for the F²MC-16LX family.

The check tool searches the code section in the absolute format load module file (ABS file) made by the SOFTUNE V3 tool and detects an instruction that seems to access the low power consumption mode control register (LPMCR: address A0). The tool checks an instruction string following the detected instruction and outputs a warning message if the string is improper.

Check whether the instruction string for which a warning message is output is applicable to the cautions in **Appendix Cautions for Access to Low Power Consumption Mode Control Register (LPMCR) for Standby Mode Transition**. If applicable, change the instruction string to one given in the cautions. No warning message will be issued for the standby mode transition instructions that meet the cautions.

This check tool statistically checks a customer program. Therefore, it may not completely detect any code and instruction that seem to access the low power consumption mode control register (LPMCR: address A0), if the customer program performs a transition to standby mode using an indirect register at runtime. In such a case, user must check the standby mode transition instruction strings.

For details about the general checking process of this checking tool and detectable standby transition instructions, see section 5. Checking Methods and section 6 Detectable Standby Transition Instructions.

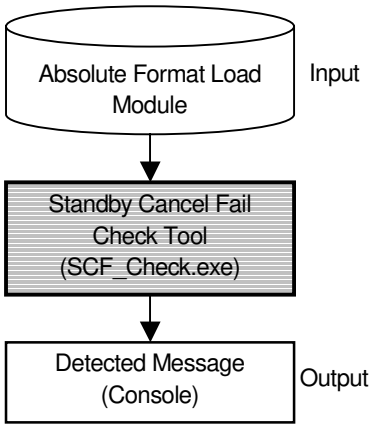


Fig. 1.1 Function of Standby Cancel Fail Check Tool

2 How to Operate the Check Tool

This section explains how to startup the check tool and its options.

How to Startup

Use the following procedures to operate the check tool.

[Format]

```
SCF_Check [<Options>] <Absolute Format Load Module Filename>
```

The default extension of an absolute format load module file is “ABS.”

[Example]

```
SCF_Check module.abs
SCF_Check –Wf module.abs
```

Options

The following shows an option for the check tool.

Option Name	Explanation
-Wf	Checks only in fetch units. Does not check in byte units.

Destination of Warning Messages

Check tool warning messages are output to standard outputs.
To record warning messages to files, use the OS re-direct function.

[Example]

```
SCF_Check >check.log module.abs
SCF_Check >check.log –Wf module.abs
```

How to Use on the Command Line

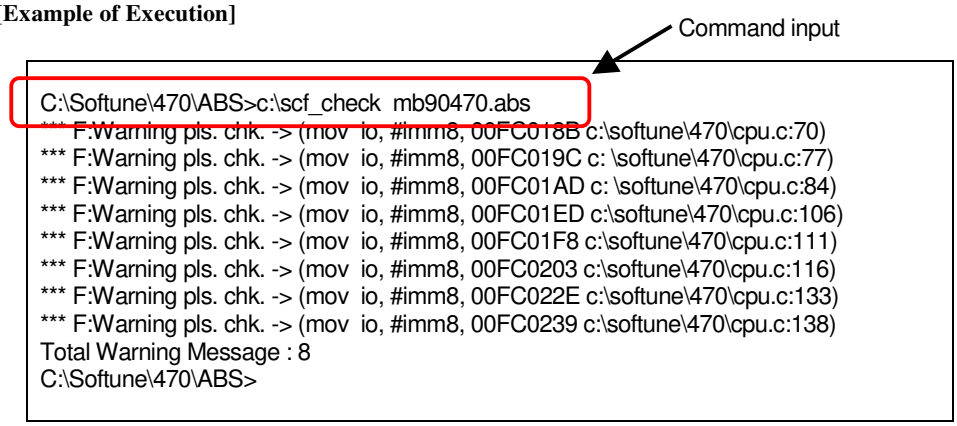
1. Copy SCF_Check.exe to any folder.
2. Start up the MS-DOS prompt (command prompt).
3. Move it to the folder that has the absolute format load module of the target for checking, then input the command as follows.

```
Full path name of the folder to where SCF_Check.exe was copied\SCF_Check
                    absolute format load module name
```

[Example]

```
C:\SCF\SCF_Check  module.abs
C:\SCF\SCF_Check  -Wf module.abs
```

[Example of Execution]

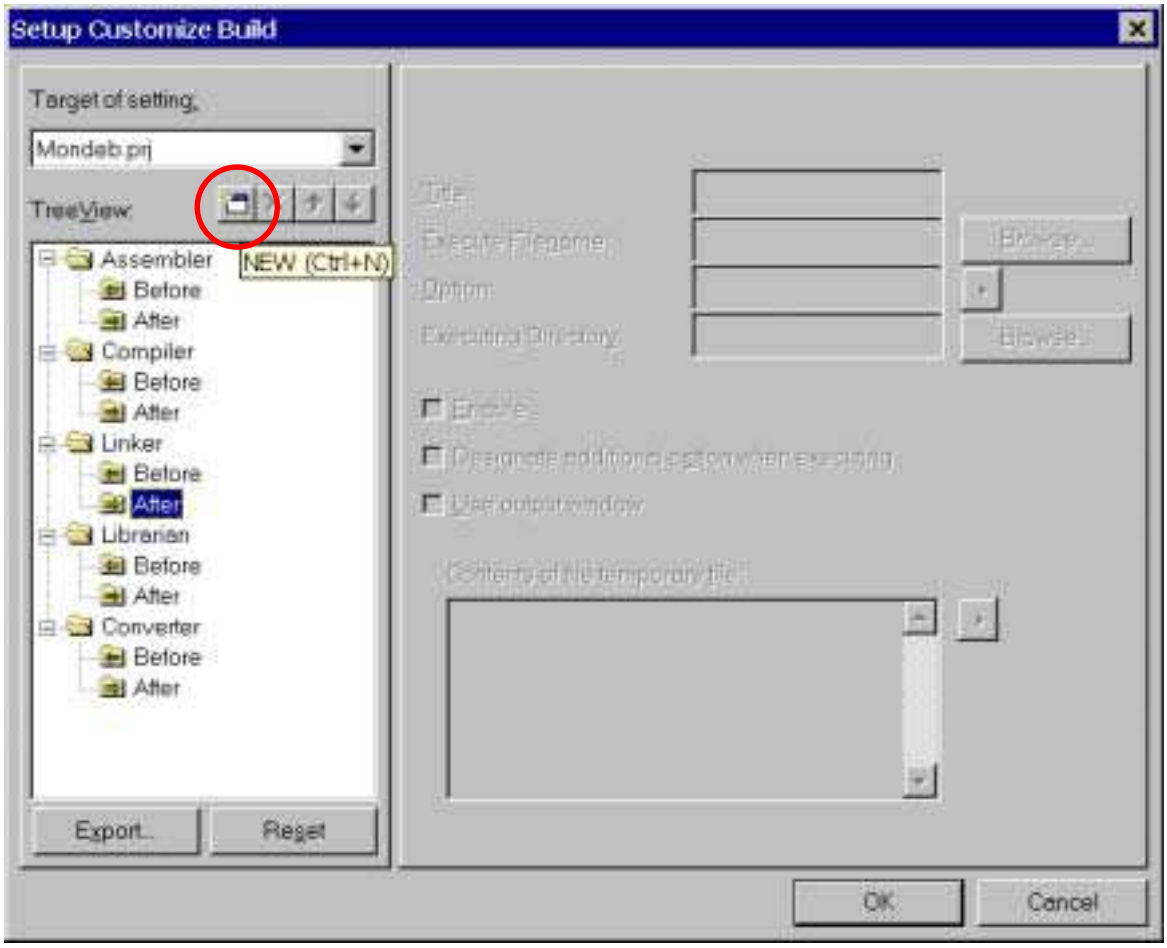


How to Register to the Workbench Customize Build Function

If you use a SOFTUNE Workbench in the V30L25 or later version register SCF_Check.exe to the customize build function, so that the automatical check is performed after creating a load module when executing Build or Make, using the following procedures.

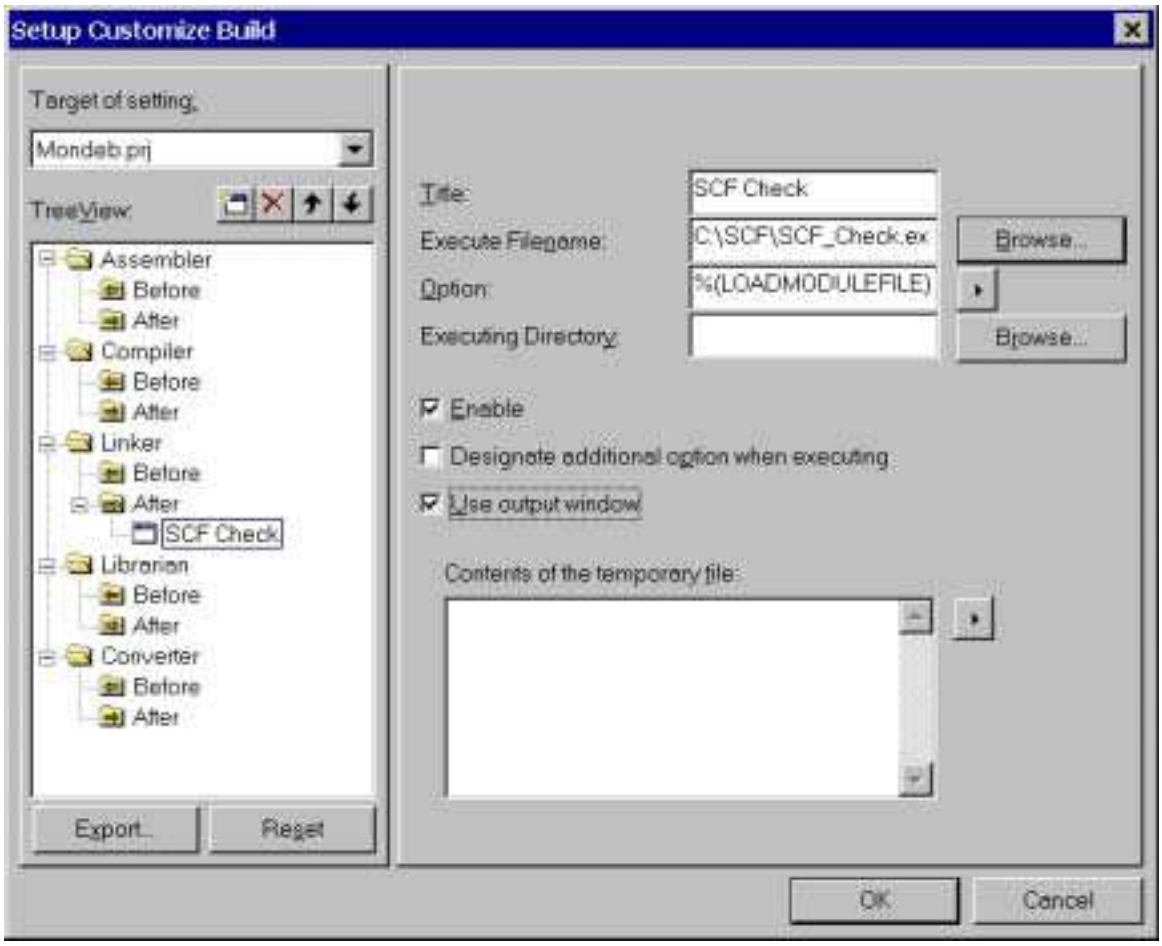
How to Register

1. Copy SCF_Check.exe to any folder.
2. Start up SOFTUNE Workbench and open the check target project.
3. Click [Setting Customize Build] in the SOFTUNE Workbench [Project] enu.
4. Select [After] in the [Linker], in the [Setup Customize Build] dialog box, and then click [New].



5. Input the following to [Title], [Execute Filename] and [Option] in the [Setup Customize Build] dialog box, and then check [Enable] and [Use output window].

	Input Example	Remarks
Title	SCF Check	
Execute Filename	C:\SCF\SCF_Check.exe	Specify check tools using full path
Option	%(LOADMODULEFILE)	Specify load module file. (Specify load module file -> full path name)
Executing Directory		Not necessary.

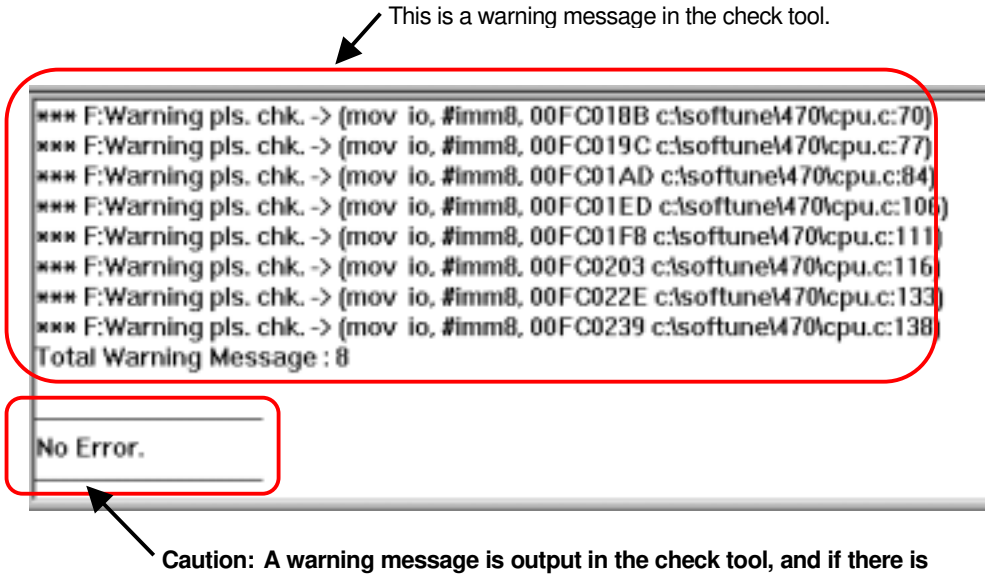


6. Click [OK] to end the [Setup Customize Build].

How to Use

1. Executing Make or Build executes the check tool automatically. If a warning message is issued, the following message is output to the output window.

[Example of Execution]



Caution: A warning message is output in the check tool, and if there is no error, “No Error” is output.

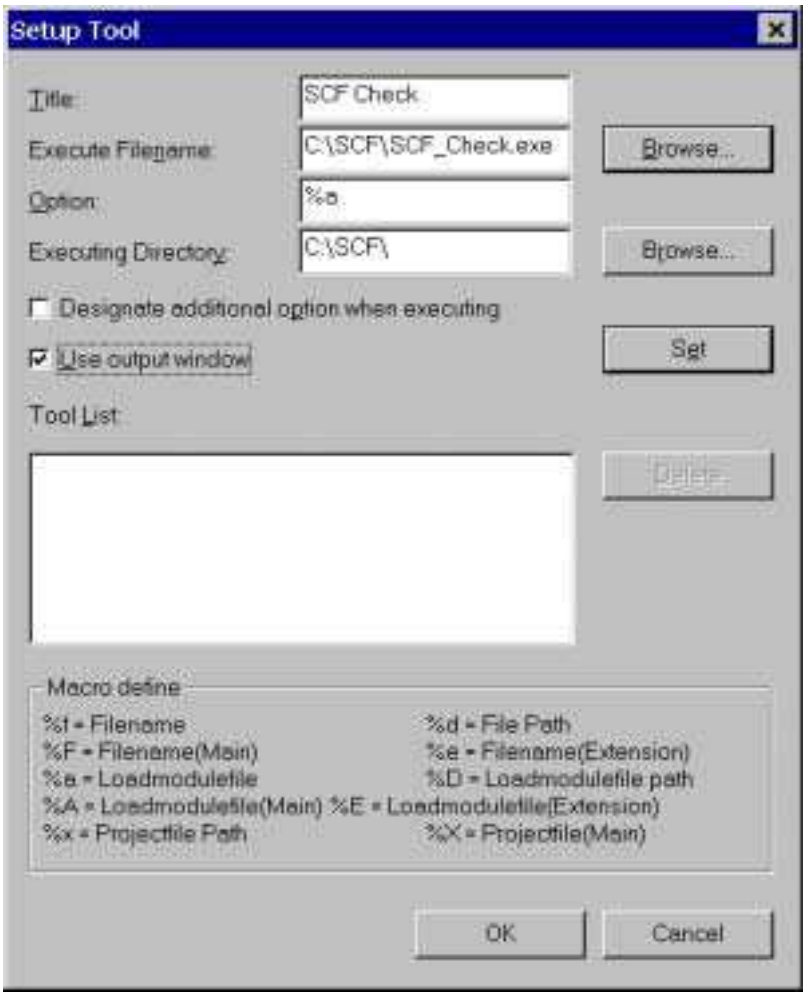
How to Register in the Startup of the SOFTUNE Workbench Tools and How to Use

If you are using a SOFTUNE Workbench in the V30L24 or later version, registers SCF_Check.exe in [Start Tool] in [Tool], to use check tools from the Wrokbench, using the following procedures.

How to Register

1. Copy SCF_Check.exe to any folder.
2. Start up the Softune Workbench.
3. Click [Tool] in the SOFTUNE Workbench [Setup] menu.
4. Input the following to [Title], [Execute Filename] and [Option] in the [Setup Tool] dialog box, and then mark [Use output window].

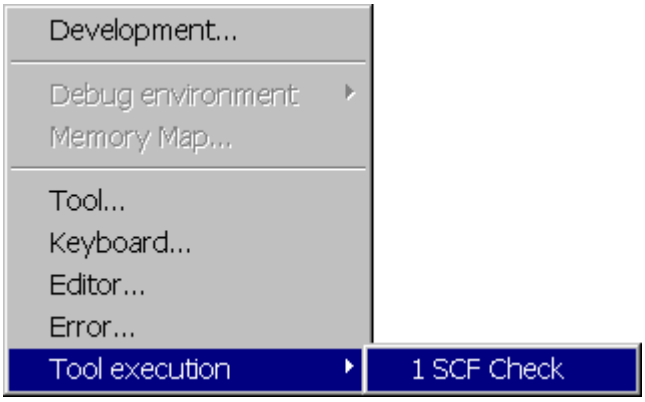
	Input Example	Remarks
Title	SCF Check	
Execute Filename	C:\SCF\SCF_Check.exe	Specify check tools using full path
Option	%a	Specify load module file.
Executing Directory	C:\SCF	Set [Execute Filename] for automatical setting.



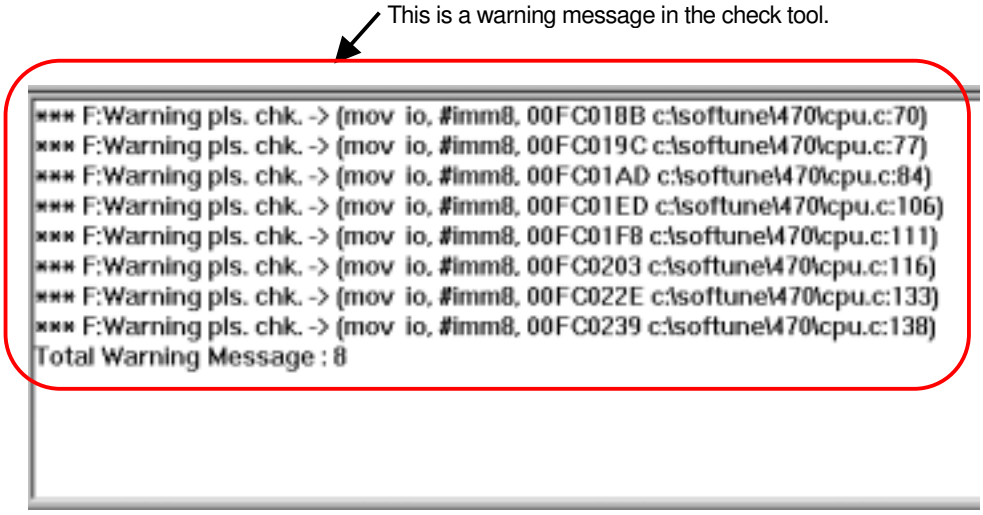
5. Click [Set] in the [Setup Tool] dialog box to register the tools.
6. Click [OK] to end the [Setup Tool].

How to Use

1. Select the title name set by "Setup Tool" in the [Start Tool] in the SOFTUNE Workbench [Setup] menu.



[Example of Execution]



3 Checking Procedures

This section explains how to check using the check tool.

Checking Procedures

Use the check tool to check using the following procedures.

Procedure 1 Applying the check tool without specifying the option.

Redundant messages may be output, but checks both in fetch unit and in byte unit to detect all suspicious locations.
This provides a complete, no-miss, check.
(However, this excludes indirect register detections when executing.)

[Example]

```
SCF_Check module.abs
```

Procedure 2 Reviewing locations where warning messages were issued and amending locations that require amendments.

Ignore redundant warning messages.

Procedure 3 Specify the -Wf option and re-check.

Verify that there are no problems.

[Example]

```
SCF_Check –Wf  module.abs
```

Repeat procedure 2 and 3 until no more warnings are issued.

4 Warning Messages

This section explains the warning messages output when the standby mode transition instruction strings are detected that may not meet Cautions for Access to Low Power Consumption Mode Control Register (LPMCR) for Standby Mode Transition.

Warning Message

This check tool outputs the following message when it detects the standby mode transition instruction strings that do not meet Cautions for Access to Low Power Consumption Mode Control Register (LPMCR) for Standby Mode Transition.

[Format]

Warning Message (Instruction Pattern Absolute Address Source File Name: Line Number)

In the warning message field, of the two check methods of the fetch unit and byte unit checks, either an F or a B is applied after '***' to identify by which check the instruction pattern was found.

[Example of By a Fetch Unit Check]

*** F:Warning pls. chk. -> (setb io,#imm8 0x00ffffff source filename:line number)

[Example of By a Byte Unit Check]

*** B:Warning pls. chk. -> (setb io,#imm8 0x00ffffff source filename:line number)

[Remarks]

The source filename, line number as auxiliary information in the parenthesis indicates it was received from the debug information.
If there is no debug information, the source filename and line number do not appear.
To show the source filename and line number, rebuild the program targeted for the check with the debug information.

5 Check method

This section explains how to check.

Checking method

This tool uses the two methods of checking program, one in **a fetch unit** and the other in **a byte unit**.

Fetch unit check

Fetches instructions and reads each instruction to check.
With the fetch unit check, it checks on instruction units so there is no possibility for redundant messages to be issued by mis-detection. However, the back side to this is that accurate checks may not be possible when data other than instructions are output to the code section.

[Example]

Code Section Byte Image

0x42	0x6C	0x43	0xA0	...

Operation Code, Operand (Hex)	Instruction
0x42, 0x6C	MOV A,#0x6C
0x43, 0xA0	MOVX A,#0xA0
. . .	

Checks MOV A,#0x6C.
Checks MOVX A,#0xA0.
.
.
.

When this tool detects the standby mode transition instruction strings that do not meet Cautions for Access to Low Power Consumption Mode Control Register (LPMCR) for Standby Mode Transition in a fetch unit check, it will output the following message.

*** F:Warning pls. chk. -> (movw io,A, 00F90003 module.c:61)

Byte unit check

This checks using single byte data units regardless of the instruction.
Because this checks using byte patterns while updating the address for single bytes, the byte unit check does not allow miss-detection. The back side to this is that if it determines an operand to be an operation code, there is the possibility that it will output redundant warning messages.
Since a redundant warning message may be output, the programmer should check whether the instruction string for which a warning message is output is a real standby mode transition instruction string.

[Example]

Code Section Byte Image

0x42	0x6C	0x43	0xA0	...

Operation Code, Operand (Hex)	Instruction
0x42, 0x6C	MOV A,#0x6C
0x43, 0xA0	MOVX A,#0xA0
. . .	

Checks 0x42.
Checks 0x6C.
Checks 0x43.
Checks 0xA0.
.
.
.

When this tool detects the standby mode transition instruction strings that do not meet Cautions for Access to Low Power Consumption Mode Control Register (LPMCR) for Standby Mode Transition in a byte unit check, it will output the following message.

*** B:Warning pls. chk. -> (movw io,A, 00F90003 module.c:61)

When Warning Messages Are Not Output Even When the Standby Transition Instruction is Detected

Even if this tool check detects a standby mode transition instruction, it will output no warning message when an instruction following the standby mode transition instruction is the following instruction string.

1. **When an instruction in Table 3 in Appendix immediately after a standby transition instruction is described.**

[Example]

```
MOV     LPMCR,#xxh      ; Standby transition instruction
CLRB    EIRR:0          ; Instruction in Table 3 in Appendix
MOV     A,#010H         ; Any instruction
          .
          .
          .
```

For details about instructions that do not update the queue counter, refer to F²MC-16LX Standby Cancel Fail in the Customer Notification Report.

2. **When the measure code is described immediately after a standby transition instruction.**

Measure Code: JMPP, JCTX, RET, RETP, RETI instructions, describing at least two NOP instruction and JMP system instruction immediately after the standby transition instruction are acceptable.
Excluding JMP @(@PC+disp16),JMPP @(@PC+disp16).

[Example]

```
MOV     LPMCR,#xxh      ; Standby transition instruction
NOP                     ; Measure Code   First NOP
NOP                     ; Measure Code   Second NOP
JMP     LABEL           ; Measure Code   JMP system instruction
MOV     A,#010H         ; Any instruction
          .
          .
          .
```

6 Detectable Standby Mode Transition Instructions

This section explains detectable standby mode transition instructions.

Detectable Standby Mode Transition Instructions

If any of the following instructions is used as a standby mode transition instruction, this tool checks instructions following the instruction and outputs a warning message.

Table 6a Standby Transition Instruction 1

Instruction		Remarks
mov	io, #imm8	Detects when io,dir,addr16 is 0x00A0 and #imm8, #imm16 bit4 is 1 and bit7 is 1 or bit6 is 1 or bit3 is 0.
mov	dir,#imm8	
movw	io,#imm16	
mov	addr16,#imm8	
movw	addr16,#imm16	

Table 6b Standby Transition Instruction 2

Instruction		Remarks
setb	io:bp	Detects when io,dir,addr16 is 0x00A0 and bp is 7 or 6.
setb	dir:bp	
setb	addr16:bp	

Table 6c Standby Transition Instruction 3

Instruction		Remarks
clrb	io:bp	Detects when io,dir,addr16 is 0x00A0 and bp is 3.
clrb	dir:bp	
clrb	addr16:bp	

Table 6d Standby Transition Instruction 4

Instruction		Remarks
mov	io,A	Detects when io,dir,addr16 is 0x00A0.
mov	dir,A	
mov	addr16,A	
movw	io,A	
movw	dir,A	
movw	addr16,A	
or	addr16,A	
and	addr16,A	
orw	addr16,A	
andw	addr16,A	

Table 6e Standby Transition Instruction 5

Instruction		Remarks
movb	io:bp,A	Detects when io,dir,addr16 is 0x00A0 and bp is 3 or 6 or 7.
movb	dir:bp,A	
movb	addr:bp,A	

Note:

This check tool only performs static checks. It detects only instructions that can be clearly regarded as writing to the low power consumption mode control register (LPMCR/address 0x00A0).

In a program that a transition to standby mode is performed by indirect addressing with an enabled register in which the address (0x00A0) of the low power consumption mode control register is set, the check tool cannot detect any standby mode transition instruction.

For details about detectable standby transition instructions, see the explanation on detectable standby transition instructions in section 7 Restrictions.

7 Restrictions

This section explains restrictions on the check tool.

Restrictions

This check tool has the following restrictions.

Table 7 Restrictions

Items	Restrictions
Undetectable Standby Transition Instructions	This check tool detects standby mode transition instructions only when a transition to standby mode is performed by the instructions shown in Section 6 Detectable Standby mode Transition Instructions. Detection is impossible when performing a standby transition using the following instructions. **MOV eam, #imm8** **MOV eam, Ri** **MOV eam, A** **MOV @RLi+disp8, A** **MOVW eam, RWi** **MOVW eam, A** **MOVW @RLi+disp8, A**
Patterns outputting warnings even for avoidable Standby Cancel Fail	This check tool requires the customer to make the final determination because warnings are not issued for the following cases when a Standby Cancel Fail can be avoided. 1. Standby mode cancel interrupt does not occur in a fixed period immediately after the standby mode transition instruction is executed. 2. When an instruction is executed by the standby mode cancel interrupt there is an instruction that is not prohibited acceptance of the interrupt, because there is no data access or address generated immediately after the standby mode transition instruction. 3. When the next instruction after the standby mode transition instruction is executed by the standby mode cancel interrupt, the following conditions are met; Condition: There are at least two NOP immediately after the transition instruction and the JMP, JMPP, JCTX, RET, RETP, RETI instructions exist before the instruction using the PC value. However, interrupts are not generated until the JMP, JMPP, JCTX, RET, RETP, RETI instructions are executed. For details about avoidable Standby Cancel Fail, refer to **F²MC-16LX Standby Cancel Fail in the Customer Notification Report**.
Checkable files	This check tool targets only absolute format load modules created by the Softune V3 tool. This does not target absolute format load modules created by the earlier version Softune V1 tool.

8 Check Tool Output Error Messages

This section explains error messages that this check tool outputs.

Error Messages

This check tool detects the following errors.

1. Open error

[Format]

```
ERROR: Filename open error
```

[Example]

ERROR: module.abs open error

[Explanation]

This error is output when the targeted file cannot be opened.

2. Read error

[Format]

```
ERROR: ABS File Read Error
```

[Example]

ERROR: ABS File Read Error

[Explanation]

This error is output when the targeted file is not a Softune V3 format load module and the file cannot be read

Appendix Cautions for Access to Low Power Consumption Mode Control Register (LPMCR) for Standby Mode Transition

Cautions for Access to Low Power Consumption Mode Control Register (LPMCR)

Devices applicable to the cautions

Devices whose latest manual and its list of corrigenda list Cautions for Access to Low Power Consumption Mode Control Register (LPMCR) for Standby Mode Transition

Access to low power consumption mode control register (LPMCR) using assembler language

- Use the instructions in Table 1 when setting transition to standby mode in the low power consumption mode control register (LPMCR).
- Be sure to place an instruction string in shown below immediately after the standby mode transition instruction in Table 1. If standby control macro instructions in Table 2 are used, the following instruction string is placed. It is therefore recommendable to use standby control macro instructions when developing a program.

```
MOV   LPMCR, #H'xx     ; Low power consumption mode transition instruction in Table 1
NOP
NOP
JMP   $+3              ; Jump to next instruction
MOV   A, #H'10         ; Any instruction
```

When instruction strings other than one in is placed, no operation performed after the standby mode is canceled will be ensured.
The operation performed after the standby mode is canceled will be ensured only when any instruction in Table 3 is immediately after the standby mode transition instruction, even if the instruction string in is not placed.

Table 1 List of Instructions for Transition to Low Power Consumption Mode

MOV	io, #imm8	MOV	dir, #imm8	MOV	eam, #imm8	MOV	eam, Ri
MOV	io, A	MOV	dir, A	MOV	addr16, A	MOV	eam, A
MOV	@Rli+disp8,A						
MOVW	io, #imm16	MOVW	dir, #imm16	MOVW	eam, #imm16	MOVW	eam, RWi
MOVW	io, A	MOVW	dir, A	MOVW	addr16, A	MOVW	eam, A
MOVW	@Rli+disp8,A						
SETB	io:bp	SETB	dir:bp	SETB	addr16:bp		
CLRB	io:bp	CLRB	dir:bp	CLRB	addr16:bp		

Standby control macro instructions

When the following standby control macro instructions are used, the instruction string meets Cautions for Access to Low Power Consumption Mode Control Register (LPMCR) for Standby Mode Transition. It is therefore recommendable to use standby control macro instructions when developing a program.

Table 2 Standby Control Macro Instructions

IO_STOP_HOLD	Sets the SLP bit of the low power consumption mode control register (LPMCR) to 0 and the STP bit to 1. A transition to stop mode or pseudo clock mode is performed. Keeps the level of the external pin.
IO_STOP_Z	Sets the SLP bit of the low power consumption mode control register (LPMCR) to 1 and the STP bit to 1. A transition to stop mode or pseudo clock mode is performed. Sets the level of the external pin to high-impedance.
IO_SLEEP	Sets the SLP bit of the low power consumption mode control register (LPMCR) to 1. A transition to sleep mode is performed.
IO_TMD_HOLD	Sets the SLP bit of the low power consumption mode control register (LPMCR) to 0 and the STP bit to 1. A transition to clock mode is performed. Keeps the level of the external pins.
IO_TMD_Z	Sets the SLP bit of the low power consumption mode control register (LPMCR) to 1 and the STP bit to 1. A transition to clock mode is performed. Sets the level of the external pin to high-impedance.
IO_SET_LPMCR IMM	A value specified in IMM is set in the low power consumption mode control register (LPMCR).

Table 3 List of Instructions

ADDC A	XCHW A,@RWj	CMPW A,@RWj	RORC,A
ADDDC A	XCHW A,@RWj+	CMPW A,@RWj+	RORC @RWj
SUBC A	XCHW RWi,@RWj	CMPL A,@RWj	RORC @RWj+
SUBDC A	XCHW RWi,@RWj+	CMPL A,@RWj+	ROLC,A
ADDW A	MOVL A,@RWj	MULU A	ROLC @RWj
SUBW A	MOVL A,@RWj+	MULU A,@RWj	ROLC @RWj+
DIVU A	MOVL @RWj,A	MULU A,@RWj+	JMP @(@RWj)
MULU A	MOVL @RWj+,A	MULUW A	JMP @(@RWj+)
MULUW A	ADD A,@RWj	MULUW A,@RWj	JMPP @(@RWj)
NOT A	ADD A,@RWj+	MULUW A,@RWj+	JMPP @(@RWj+)
ANDW A	ADD @RWj,A	DIVU A,@RWj	CALL @(@RWj)
ORW A	ADD @RWj+,A	DIVU A,@RWj+	CALL @(@RWj+)
XORW A	ADDC A,@RWj	DIVUW A,@RWj	CALLP @(@RWj)
NOTW A	ADDC A,@RWj+	DIVUW A,@RWj+	CALLP @(@RWj+)
NEG A	ADDC A,ear	AND A,@RWj	CALLP @ear
NEGW A	SUB A,@RWj	AND A,@RWj+	DBNZ @RWj,REL
ASRW A	SUB A,@RWj+	AND @RWj,A	DBNZ @RWj+,REL
LSRW A / SHRW A	SUB @RWj,A	AND @RWj+,A	DWBNZ @RWj,REL
LSLW A / SHLW A	SUB @RWj+,A	OR A,@RWj	DWBNZ @RWj+,REL
SWAP	SUBC A,@RWj	OR A,@RWj+	INT9
SWAPW / XCHW A,T	SUBC A,@RWj+	OR @RWj,A	PUSHW A
EXTW	SUBC A,ear	OR @RWj+,A	PUSHW AH
MOV A,Ri	ADDW A,@RWj	XOR A,@RWj	PUSHW PS
MOV A,ear	ADDW A,@RWj+	XOR A,@RWj+	AND CCR,#imm8
MOV A,@RWj	ADDW @RWj,A	XOR @RWj,A	OR CCR,#imm8
MOV A,@RWj+	ADDW @RWj+,A	XOR @RWj+,A	MOVEA RWi,@RWj
MOVX A,Ri	ADDCW A,@RWj	NOT @RWj	MOVEA RWi,@RWj+
MOVX A,ear	ADDCW A,@RWj+	NOT @RWj+	MOVEA A,@RWj
MOVX A,@RWj	ADDCW A,ear	ANDW A,@RWj	MOVEA A,@RWj+
MOVX A,@RWj+	SUBW A,@RWj	ANDW A,@RWj+	SETB io:bp
MOV @RWj,A	SUBW A,@RWj+	ANDW @RWj,A	CLRB io:bp
MOV @RWj+,A	SUBW @RWj,A	ANDW @RWj+,A	CBNE @RWj,#imm8,rel
MOV Ri,@RWj	SUBW @RWj+,A	ORW A,@RWj	CWBNE
MOV Ri,@RWj+	SUBCW A,@RWj	ORW A,@RWj+	@RWj,#imm16,rel
MOV @RWj,Ri	SUBCW A,@RWj+	ORW @RWj,A	MUL A,ear
MOV @RWj+,Ri	SUBCW A,ear	ORW @RWj+,A	MUL A,@RWj
MOV @RWj,#imm8	ADDL A,@RWj	XORW A,@RWj	MUL A,@RWj+
MOV @RWj+,#imm8	ADDL A,@RWj+	XORW A,@RWj+	MULW A,RWi
XCH A,@RWj	SUBL A,@RWj	XORW @RWj,A	MULW A,@RWj
XCH A,@RWj+	SUBL A,@RWj+	XORW @RWj+,A	MULW A,@RWj+
XCH Ri,@RWj	INC @RWj	NOTW @RWj	DIV A,@RWj
XCH Ri,@RWj+	INC @RWj+	NOTW @RWj+	DIV A,@RWj+
MOVW A,RWi	DEC @RWj	ANDL A,@RWj	DIVW A,@RWj
MOVW A,ear	DEC @RWj+	ANDL A,@RWj+	DIVW A,@RWj+
MOVW A,@RWj	INCW @RWj	ORL A,@RWj	MUL A
MOVW A,@RWj+	INCW @RWj+	ORL A,@RWj+	MULW A
MOVW @RWj,A	DECW @RWj	XORL A,@RWj	DIV A
MOVW @RWj+,A	DECW @RWj+	XORL A,@RWj+	
MOVW RWi,@RWj	INCL @RWj	NEG @RWj	
MOVW RWi,@RWj+	INCL @RWj+	NEG @RWj+	
MOVW @RWj,RWi	DECL @RWj	NEGW A	
MOVW @RWj+,RWi	DECL @RWj+	NEGW @RWj	
MOVW @RWj,#imm16	CMP A,@RWj	NEGW @RWj+	
MOVW @RWj+,#imm16	CMP A,@RWj+	NRML A,R0	

Access to low power consumption mode control register (LPMCR) using C language, not standby control macro instructions

To set transition to standby mode in the low power consumption mode control register (LPMCR), use any of the following methods:

1. Use an instruction for transition to standby mode as a function, and insert two built-in instructions of _wait_nop() immediately after the standby mode transition instruction. If interrupts other than one for return to standby mode may occur within the function, perform optimization at compiling time and inhibit the occurrence of a LINK/UNLINK instruction.

   Example (Function for transition to clock mode or time-based timer mode)

```
void enter_watch() {
  IO_LPMCR.byte = 0x10;  /* Set TMD bit of LPMCR to 0. */
  __wait_nop();
  __wait_nop();
}
```

2. Write an instruction for transition to standby mode in the _asm statement, and insert two NOP and JMP instructions immediately after the standby mode transition instruction.

   Example (Transition to sleep mode)

```
__asm("       MOV   I:_IO_LPMCR, #H'58");     /* Set SLP of LPMCR to 1. */
__asm("       NOP");
__asm("       NOP");
__asm("       JMP    $+3");                   /* Jump to next instruction. */
```

3. Write an instruction for transition to standby mode between #pragma asm and #pragma endasm, and insert two NOP and JMP instructions immediately after the standby mode transition instruction.

   Example (Transition to stop mode)

```
#pragma asm
        MOV     I:_IO_LPMCR, #H'98   /* Set STP of LPMCR to 1. */
        NOP
        NOP
        JMP     $+3                  /* Jump to next instruction. */
#pragma endasm
```