

Polyspace Bug Finder

Detailed Report for Project: Ilcc68

Report Author: LibDriver

Polyspace Bug Finder: Detailed Report for Project: Ilcc68

by Report Author: LibDriver

Published 02-Apr-2023 22:33:17

Analysis Author(s): LibDriver

Polyspace Version(s): Polyspace Bug Finder 3.2 (R2020a)

Project Version(s): 1.0

Result Folder(s):

E:\Polyspace\Ilcc68\Module\BF_Result

Table of Contents

Chapter 1. Polyspace Bug Finder Summary	1
Chapter 2. MISRA C:2012 Guidelines	2
MISRA C:2012 Guidelines Summary - Violations by File	2
MISRA C:2012 Guidelines Violations	2
Chapter 3. Defects	72
Defects	72
Chapter 4. Appendix 1 - Configuration Settings	73
Polyspace Settings	73
Coding Standard Configuration	74
Chapter 5. Appendix 2 - Definitions	82

Chapter 1. Polyspace Bug Finder Summary

Table 1.1. Project Summary

	Count	Reviewed	Unreviewed	Pass/Fail
MISRA C:2012 Guidelines	692	692	0	Pass
Defects	0	0	0	Pass
Total	692	692	0	Pass

Table 1.2. Summary By File

File	Defects (Reviewed)	MISRA C:2012 Guidelines (Reviewed)
E:\Github\llcc68\example\driver_llcc68_lora.c	0 (0)	72 (72)
E:\Github\llcc68\example\driver_llcc68_lora.h	0 (0)	0 (0)
E:\Github\llcc68\interface\driver_llcc68_interface.h	0 (0)	0 (0)
E:\Github\llcc68\interface\driver_llcc68_interface_template.c	0 (0)	21 (21)
E:\Github\llcc68\src\driver_llcc68.c	0 (0)	283 (283)
E:\Github\llcc68\src\driver_llcc68.h	0 (0)	5 (5)
E:\Github\llcc68\test\driver_llcc68_cad_test.c	0 (0)	32 (32)
E:\Github\llcc68\test\driver_llcc68_cad_test.h	0 (0)	0 (0)
E:\Github\llcc68\test\driver_llcc68_register_test.c	0 (0)	182 (182)
E:\Github\llcc68\test\driver_llcc68_register_test.h	0 (0)	0 (0)
E:\Github\llcc68\test\driver_llcc68_sent_receive_test.c	0 (0)	97 (97)
E:\Github\llcc68\test\driver_llcc68_sent_receive_test.h	0 (0)	0 (0)

Chapter 2. MISRA C:2012 Guidelines

MISRA C:2012 Guidelines Summary - Violations by File

File	Total
E:\Github\llcc68\example\driver_llcc68_lora.c	72
E:\Github\llcc68\interface\driver_llcc68_interface_template.c	21
E:\Github\llcc68\src\driver_llcc68.c	283
E:\Github\llcc68\src\driver_llcc68.h	5
E:\Github\llcc68\test\driver_llcc68_cad_test.c	32
E:\Github\llcc68\test\driver_llcc68_register_test.c	182
E:\Github\llcc68\test\driver_llcc68_sent_receive_test.c	97
Total	692

MISRA C:2012 Guidelines Violations

Table 2.1. E:\Github\llcc68\example\driver_llcc68_lora.c

ID	Guideline	Message	Function	Severity	Status	Comment
637	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
507	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
482	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
438	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
423	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
407	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.

368	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
362	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
421	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
530	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
419	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
349	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
453	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
634	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
629	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
610	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
422	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
402	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
545	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
303	10.1	Operands shall not be of an inappropriate essential type. The right operand of the = operator is of an inappropriate essential type category signed.	llcc68_lora_init()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
379	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.

427	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
374	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
406	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
298	10.1	Operands shall not be of an inappropriate essential type. The right operand of the = operator is of an inappropriate essential type category signed.	llcc68_lora_init()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
412	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
266	10.1	Operands shall not be of an inappropriate essential type. The left operand of the operator is of an inappropriate essential type category enum. The right operand of the operator is of an inappropriate essential type category enum.	llcc68_lora_set_continuous_receive_mode()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
276	10.1	Operands shall not be of an inappropriate essential type. The left operand of the operator is of an inappropriate essential type category signed. The right operand of the operator is of an inappropriate essential type category enum.	llcc68_lora_set_continuous_receive_mode()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
290	10.4	Both operands of an operator in which the usual arithmetic conversions are performed shall have the same essential type category. The left operand of the operator has essentially signed type while the right operand has essentially enum type.	llcc68_lora_set_continuous_receive_mode()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
269	10.4	Both operands of an operator in which the usual arithmetic conversions are performed shall have the same essential type category. The left operand of the operator has essentially signed type while the right operand has essentially enum type.	llcc68_lora_set_continuous_receive_mode()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
274	10.1	Operands shall not be of an inappropriate essential type. The left operand of the operator is of an inappropriate essential type category signed.	llcc68_lora_set_continuous_receive_mode()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers

		The right operand of the operator is of an inappropriate essential type category enum.				guarantee the safety of the operation.
285	10.1	Operands shall not be of an inappropriate essential type. The left operand of the operator is of an inappropriate essential type category signed. The right operand of the operator is of an inappropriate essential type category enum.	llcc68_lora_set_continuous_receive_mode()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
301	10.4	Both operands of an operator in which the usual arithmetic conversions are performed shall have the same essential type category. The left operand of the operator has essentially signed type while the right operand has essentially enum type.	llcc68_lora_set_continuous_receive_mode()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
280	10.1	Operands shall not be of an inappropriate essential type. The left operand of the operator is of an inappropriate essential type category enum. The right operand of the operator is of an inappropriate essential type category enum.	llcc68_lora_set_continuous_receive_mode()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
278	10.4	Both operands of an operator in which the usual arithmetic conversions are performed shall have the same essential type category. The left operand of the operator has essentially signed type while the right operand has essentially enum type.	llcc68_lora_set_continuous_receive_mode()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
289	10.1	Operands shall not be of an inappropriate essential type. The left operand of the operator is of an inappropriate essential type category signed. The right operand of the operator is of an inappropriate essential type category enum.	llcc68_lora_set_continuous_receive_mode()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
267	10.4	Both operands of an operator in which the usual arithmetic conversions are performed shall have the same essential type category. The left operand of the operator has essentially signed type while the right operand has essentially enum type.	llcc68_lora_set_continuous_receive_mode()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
270	10.1	Operands shall not be of an inappropriate essential type. The left operand of the operator is of an inappropriate essential type category signed. The right operand of the operator is of an inappropriate essential type category enum.	llcc68_lora_set_continuous_receive_mode()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.

262	10.4	Both operands of an operator in which the usual arithmetic conversions are performed shall have the same essential type category. The left operand of the operator has essentially signed type while the right operand has essentially enum type.	llcc68_lora_set_continuous_receive_mode()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
264	10.1	Operands shall not be of an inappropriate essential type. The left operand of the operator is of an inappropriate essential type category signed. The right operand of the operator is of an inappropriate essential type category enum.	llcc68_lora_set_continuous_receive_mode()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
1	20.9	All identifiers used in the controlling expression of #if or #elif preprocessing directives shall be #define'd before evaluation. 'LLCC68_BOOL_FALSE' is not defined.	File Scope	Low	Justified	Have defined.
265	10.4	Both operands of an operator in which the usual arithmetic conversions are performed shall have the same essential type category. The left operand of the = operator has essentially unsigned type while the right operand has essentially signed type.	llcc68_lora_set_continuous_receive_mode()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
272	10.1	Operands shall not be of an inappropriate essential type. The right operand of the = operator is of an inappropriate essential type category signed.	llcc68_lora_set_continuous_receive_mode()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
271	10.1	Operands shall not be of an inappropriate essential type. The left operand of the << operator is of an inappropriate essential type category signed.	llcc68_lora_set_continuous_receive_mode()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
304	10.1	Operands shall not be of an inappropriate essential type. The left operand of the operator is of an inappropriate essential type category enum. The right operand of the operator is of an inappropriate essential type category enum.	llcc68_lora_set_shot_receive_mode()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
263	10.4	Both operands of an operator in which the usual arithmetic conversions are performed shall have the same essential type category. The left operand of the operator has essentially signed type while the	llcc68_lora_set_shot_receive_mode()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of

		right operand has essentially enum type.				the operation.
283	10.1	Operands shall not be of an inappropriate essential type. The left operand of the operator is of an inappropriate essential type category signed. The right operand of the operator is of an inappropriate essential type category enum.	llcc68_lora_set_shot_receive_mode()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
273	10.4	Both operands of an operator in which the usual arithmetic conversions are performed shall have the same essential type category. The left operand of the operator has essentially signed type while the right operand has essentially enum type.	llcc68_lora_set_shot_receive_mode()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
286	10.1	Operands shall not be of an inappropriate essential type. The left operand of the operator is of an inappropriate essential type category signed. The right operand of the operator is of an inappropriate essential type category enum.	llcc68_lora_set_shot_receive_mode()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
277	10.4	Both operands of an operator in which the usual arithmetic conversions are performed shall have the same essential type category. The left operand of the operator has essentially signed type while the right operand has essentially enum type.	llcc68_lora_set_shot_receive_mode()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
302	10.1	Operands shall not be of an inappropriate essential type. The left operand of the operator is of an inappropriate essential type category signed. The right operand of the operator is of an inappropriate essential type category enum.	llcc68_lora_set_shot_receive_mode()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
279	10.1	Operands shall not be of an inappropriate essential type. The left operand of the operator is of an inappropriate essential type category enum. The right operand of the operator is of an inappropriate essential type category enum.	llcc68_lora_set_shot_receive_mode()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
281	10.4	Both operands of an operator in which the usual arithmetic conversions are performed shall have the same essential type category. The left operand of the operator has essentially signed type while the right operand has essentially enum type.	llcc68_lora_set_shot_receive_mode()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
291	10.1	Operands shall not be of an inappropriate essential type.	llcc68_lora_set_shot_receive_mode()	Low	Not a defect	Embedded drivers need

		<p>The left operand of the operator is of an inappropriate essential type category signed.</p> <p>The right operand of the operator is of an inappropriate essential type category enum.</p>				<p>this method to set or clear some bits and drivers guarantee the safety of the operation.</p>
275	10.4	<p>Both operands of an operator in which the usual arithmetic conversions are performed shall have the same essential type category.</p> <p>The left operand of the operator has essentially signed type while the right operand has essentially enum type.</p>	llcc68_lora_set_shot_receive_mode()	Low	Not a defect	<p>Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.</p>
282	10.1	<p>Operands shall not be of an inappropriate essential type.</p> <p>The left operand of the operator is of an inappropriate essential type category signed.</p> <p>The right operand of the operator is of an inappropriate essential type category enum.</p>	llcc68_lora_set_shot_receive_mode()	Low	Not a defect	<p>Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.</p>
284	10.4	<p>Both operands of an operator in which the usual arithmetic conversions are performed shall have the same essential type category.</p> <p>The left operand of the operator has essentially signed type while the right operand has essentially enum type.</p>	llcc68_lora_set_shot_receive_mode()	Low	Not a defect	<p>Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.</p>
299	10.1	<p>Operands shall not be of an inappropriate essential type.</p> <p>The left operand of the operator is of an inappropriate essential type category signed.</p> <p>The right operand of the operator is of an inappropriate essential type category enum.</p>	llcc68_lora_set_shot_receive_mode()	Low	Not a defect	<p>Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.</p>
2	20.9	<p>All identifiers used in the controlling expression of #if or #elif preprocessing directives shall be #define'd before evaluation.</p> <p>'LLCC68_BOOL_FALSE' is not defined.</p>	File Scope	Low	Justified	<p>Have defined.</p>
268	10.4	<p>Both operands of an operator in which the usual arithmetic conversions are performed shall have the same essential type category.</p> <p>The left operand of the = operator has essentially unsigned type while the right operand has essentially signed type.</p>	llcc68_lora_set_shot_receive_mode()	Low	Not a defect	<p>Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.</p>
293	10.1	<p>Operands shall not be of an inappropriate essential type.</p> <p>The right operand of the = operator is of an inappropriate essential type category signed.</p>	llcc68_lora_set_shot_receive_mode()	Low	Not a defect	<p>Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.</p>

287	10.1	Operands shall not be of an inappropriate essential type. The left operand of the << operator is of an inappropriate essential type category signed.	llcc68_lora_set_shot_receive_mode()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
292	10.1	Operands shall not be of an inappropriate essential type. The left operand of the operator is of an inappropriate essential type category enum. The right operand of the operator is of an inappropriate essential type category enum.	llcc68_lora_set_sent_mode()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
288	10.4	Both operands of an operator in which the usual arithmetic conversions are performed shall have the same essential type category. The left operand of the operator has essentially signed type while the right operand has essentially enum type.	llcc68_lora_set_sent_mode()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
305	10.1	Operands shall not be of an inappropriate essential type. The left operand of the operator is of an inappropriate essential type category signed. The right operand of the operator is of an inappropriate essential type category enum.	llcc68_lora_set_sent_mode()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
294	10.1	Operands shall not be of an inappropriate essential type. The left operand of the operator is of an inappropriate essential type category signed. The right operand of the operator is of an inappropriate essential type category enum.	llcc68_lora_set_sent_mode()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
297	10.4	Both operands of an operator in which the usual arithmetic conversions are performed shall have the same essential type category. The left operand of the operator has essentially signed type while the right operand has essentially enum type.	llcc68_lora_set_sent_mode()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
296	10.1	Operands shall not be of an inappropriate essential type. The left operand of the operator is of an inappropriate essential type category enum. The right operand of the operator is of an inappropriate essential type category enum.	llcc68_lora_set_sent_mode()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
300	10.4	Both operands of an operator in which the usual arithmetic conversions are performed shall have the same essential type	llcc68_lora_set_sent_mode()	Low	Not a defect	Embedded drivers need this method to set or clear

		category. The left operand of the operator has essentially signed type while the right operand has essentially enum type.				some bits and drivers guarantee the safety of the operation.
306	10.1	Operands shall not be of an inappropriate essential type. The left operand of the operator is of an inappropriate essential type category signed. The right operand of the operator is of an inappropriate essential type category enum.	llcc68_lora_set_sent_mode()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
295	10.4	Both operands of an operator in which the usual arithmetic conversions are performed shall have the same essential type category. The left operand of the operator has essentially signed type while the right operand has essentially enum type.	llcc68_lora_set_sent_mode()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
307	10.1	Operands shall not be of an inappropriate essential type. The left operand of the operator is of an inappropriate essential type category signed. The right operand of the operator is of an inappropriate essential type category enum.	llcc68_lora_set_sent_mode()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.

Table 2.2. E:\Github\llcc68\interface\driver_llcc68_interface_template.c

ID	Guideline	Message	Function	Severity	Status	Comment
4	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type category enum) is assigned to an object with a different essential type category (unsigned)	llcc68_interface_receive_callback()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
502	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
7	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type category enum) is assigned to an object with a different essential type category (unsigned)	llcc68_interface_receive_callback()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
358	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
6	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type category enum) is assigned to an	llcc68_interface_receive_callback()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee

		object with a different essential type category (unsigned)				the safety of the operation.
428	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
14	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type category enum) is assigned to an object with a different essential type category (unsigned)	llcc68_interface_receive_callback()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
366	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
9	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type category enum) is assigned to an object with a different essential type category (unsigned)	llcc68_interface_receive_callback()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
354	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
11	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type category enum) is assigned to an object with a different essential type category (unsigned)	llcc68_interface_receive_callback()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
359	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
12	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type category enum) is assigned to an object with a different essential type category (unsigned)	llcc68_interface_receive_callback()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
595	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
8	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type category enum) is assigned to an object with a different essential type category (unsigned)	llcc68_interface_receive_callback()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
347	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
13	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category.	llcc68_interface_receive_callback()	Low	Not a defect	Embedded drivers need this method to set or clear some

		The expression (of essential type category enum) is assigned to an object with a different essential type category (unsigned)				bits and drivers guarantee the safety of the operation.
559	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
15	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type category enum) is assigned to an object with a different essential type category (unsigned)	llcc68_interface_receive_callback()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
364	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
345	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.

Table 2.3. E:\Github\llcc68\src\driver_llcc68.c

ID	Guideline	Message	Function	Severity	Status	Comment
204	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type unsigned on 16 bits) is assigned to an object with a narrower essential type (unsigned on 8 bits)	a_llcc68_spi_read_register()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
106	10.1	Operands shall not be of an inappropriate essential type. The right operand of the & operator is of an inappropriate essential type category signed.	a_llcc68_spi_read_register()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
92	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type unsigned on 16 bits) is assigned to an object with a narrower essential type (unsigned on 8 bits)	a_llcc68_spi_read_register()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
212	10.1	Operands shall not be of an inappropriate essential type. The right operand of the & operator is of an inappropriate essential type category signed.	a_llcc68_spi_read_register()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
52	10.3	The value of an expression shall not be assigned to an object with a	a_llcc68_spi_write_register()	Low	Not a	Embedded drivers need

		narrower essential type or of a different essential type category. The expression (of essential type unsigned on 16 bits) is assigned to an object with a narrower essential type (unsigned on 8 bits)			defect	this method to set or clear some bits and drivers guarantee the safety of the operation.
83	10.1	Operands shall not be of an inappropriate essential type. The right operand of the & operator is of an inappropriate essential type category signed.	a_llcc68_spi_write_register()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
25	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type unsigned on 16 bits) is assigned to an object with a narrower essential type (unsigned on 8 bits)	a_llcc68_spi_write_register()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
30	10.1	Operands shall not be of an inappropriate essential type. The right operand of the & operator is of an inappropriate essential type category signed.	a_llcc68_spi_write_register()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
19	10.1	Operands shall not be of an inappropriate essential type. The right operand of the & operator is of an inappropriate essential type category enum.	llcc68_irq_handler()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
27	10.4	Both operands of an operator in which the usual arithmetic conversions are performed shall have the same essential type category. The left operand of the & operator has essentially unsigned type while the right operand has essentially enum type.	llcc68_irq_handler()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
34	10.4	Both operands of an operator in which the usual arithmetic conversions are performed shall have the same essential type category. The left operand of the & operator has essentially unsigned type while the right operand has essentially enum type.	llcc68_irq_handler()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
237	10.1	Operands shall not be of an inappropriate essential type. The right operand of the & operator is of an inappropriate essential type category enum.	llcc68_irq_handler()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the

						safety of the operation.
40	10.1	Operands shall not be of an inappropriate essential type. The right operand of the & operator is of an inappropriate essential type category enum.	llcc68_irq_handler()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
145	10.4	Both operands of an operator in which the usual arithmetic conversions are performed shall have the same essential type category. The left operand of the & operator has essentially unsigned type while the right operand has essentially enum type.	llcc68_irq_handler()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
38	10.1	Operands shall not be of an inappropriate essential type. The right operand of the & operator is of an inappropriate essential type category enum.	llcc68_irq_handler()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
42	10.4	Both operands of an operator in which the usual arithmetic conversions are performed shall have the same essential type category. The left operand of the & operator has essentially unsigned type while the right operand has essentially enum type.	llcc68_irq_handler()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
24	10.4	Both operands of an operator in which the usual arithmetic conversions are performed shall have the same essential type category. The left operand of the & operator has essentially unsigned type while the right operand has essentially enum type.	llcc68_irq_handler()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
245	10.1	Operands shall not be of an inappropriate essential type. The right operand of the & operator is of an inappropriate essential type category enum.	llcc68_irq_handler()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
43	10.1	Operands shall not be of an inappropriate essential type. The right operand of the & operator is of an inappropriate essential type category enum.	llcc68_irq_handler()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
55	10.4	Both operands of an operator in which the usual arithmetic	llcc68_irq_handler()	Low	Not a	Embedded drivers need

		<p>conversions are performed shall have the same essential type category.</p> <p>The left operand of the & operator has essentially unsigned type while the right operand has essentially enum type.</p>			defect	this method to set or clear some bits and drivers guarantee the safety of the operation.
35	10.1	<p>Operands shall not be of an inappropriate essential type.</p> <p>The right operand of the & operator is of an inappropriate essential type category enum.</p>	llcc68_irq_handler()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
49	10.4	<p>Both operands of an operator in which the usual arithmetic conversions are performed shall have the same essential type category.</p> <p>The left operand of the & operator has essentially unsigned type while the right operand has essentially enum type.</p>	llcc68_irq_handler()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
154	10.1	<p>Operands shall not be of an inappropriate essential type.</p> <p>The right operand of the & operator is of an inappropriate essential type category enum.</p>	llcc68_irq_handler()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
183	10.4	<p>Both operands of an operator in which the usual arithmetic conversions are performed shall have the same essential type category.</p> <p>The left operand of the & operator has essentially unsigned type while the right operand has essentially enum type.</p>	llcc68_irq_handler()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
50	10.1	<p>Operands shall not be of an inappropriate essential type.</p> <p>The right operand of the = operator is of an inappropriate essential type category signed.</p>	llcc68_irq_handler()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
68	10.4	<p>Both operands of an operator in which the usual arithmetic conversions are performed shall have the same essential type category.</p> <p>The left operand of the & operator has essentially unsigned type while the right operand has essentially enum type.</p>	llcc68_irq_handler()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
115	10.1	<p>Operands shall not be of an inappropriate essential type.</p> <p>The right operand of the & operator is of an inappropriate essential type category enum.</p>	llcc68_irq_handler()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the

						safety of the operation.
218	10.4	Both operands of an operator in which the usual arithmetic conversions are performed shall have the same essential type category. The left operand of the & operator has essentially unsigned type while the right operand has essentially enum type.	llcc68_irq_handler()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
244	10.1	Operands shall not be of an inappropriate essential type. The right operand of the & operator is of an inappropriate essential type category enum.	llcc68_irq_handler()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
56	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type unsigned on 16 bits) is assigned to an object with a narrower essential type (unsigned on 8 bits)	llcc68_single_receive()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
22	10.1	Operands shall not be of an inappropriate essential type. The right operand of the & operator is of an inappropriate essential type category signed.	llcc68_single_receive()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
58	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type unsigned on 16 bits) is assigned to an object with a narrower essential type (unsigned on 8 bits)	llcc68_single_receive()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
36	10.1	Operands shall not be of an inappropriate essential type. The right operand of the & operator is of an inappropriate essential type category signed.	llcc68_single_receive()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
59	10.8	The value of a composite expression shall not be cast to a different essential type category or a wider essential type. The value of the composite expression of essential type category floating shall not be cast to the different essential type category unsigned.	llcc68_single_receive()	Low	Not a defect	We use this function to convert driver data and drivers guarantee the safety of the operation.
51	10.3	The value of an expression shall not be assigned to an object with a	llcc68_single_receive()	Low	Not a	Embedded drivers need

		narrower essential type or of a different essential type category. The expression (of essential type unsigned on 32 bits) is assigned to an object with a narrower essential type (unsigned on 8 bits)			defect	this method to set or clear some bits and drivers guarantee the safety of the operation.
60	10.1	Operands shall not be of an inappropriate essential type. The right operand of the & operator is of an inappropriate essential type category signed.	llcc68_single_receive()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
70	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type unsigned on 32 bits) is assigned to an object with a narrower essential type (unsigned on 8 bits)	llcc68_single_receive()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
69	10.1	Operands shall not be of an inappropriate essential type. The right operand of the & operator is of an inappropriate essential type category signed.	llcc68_single_receive()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
73	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type unsigned on 32 bits) is assigned to an object with a narrower essential type (unsigned on 8 bits)	llcc68_single_receive()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
255	10.1	Operands shall not be of an inappropriate essential type. The right operand of the & operator is of an inappropriate essential type category signed.	llcc68_single_receive()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
108	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type unsigned on 16 bits) is assigned to an object with a narrower essential type (unsigned on 8 bits)	llcc68_continuous_receive()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
63	10.1	Operands shall not be of an inappropriate essential type. The right operand of the & operator is of an inappropriate essential type category signed.	llcc68_continuous_receive()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the

						safety of the operation.
46	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type unsigned on 16 bits) is assigned to an object with a narrower essential type (unsigned on 8 bits)	llcc68_continuous_receive()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
75	10.1	Operands shall not be of an inappropriate essential type. The right operand of the & operator is of an inappropriate essential type category signed.	llcc68_continuous_receive()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
126	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type unsigned on 16 bits) is assigned to an object with a narrower essential type (unsigned on 8 bits)	llcc68_lora_cad()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
95	10.1	Operands shall not be of an inappropriate essential type. The right operand of the & operator is of an inappropriate essential type category signed.	llcc68_lora_cad()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
82	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type unsigned on 16 bits) is assigned to an object with a narrower essential type (unsigned on 8 bits)	llcc68_lora_cad()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
81	10.1	Operands shall not be of an inappropriate essential type. The right operand of the & operator is of an inappropriate essential type category signed.	llcc68_lora_cad()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
690	D4.14	The validity of values received from external sources shall be checked. Dereferenced pointer is from an unsecure source. Pointer may be NULL or may point to unknown memory.	llcc68_check_packet_error()	Low	Justified	(handle == NULL)checked.
88	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category.	llcc68_lora_transmit()	Low	Not a defect	Embedded drivers need this method to set or

		The expression (of essential type unsigned on 16 bits) is assigned to an object with a narrower essential type (unsigned on 8 bits)				clear some bits and drivers guarantee the safety of the operation.
31	10.1	Operands shall not be of an inappropriate essential type. The right operand of the & operator is of an inappropriate essential type category signed.	llcc68_lora_transmit()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
86	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type unsigned on 16 bits) is assigned to an object with a narrower essential type (unsigned on 8 bits)	llcc68_lora_transmit()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
84	10.1	Operands shall not be of an inappropriate essential type. The right operand of the & operator is of an inappropriate essential type category signed.	llcc68_lora_transmit()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
90	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type category enum) is assigned to an object with a different essential type category (unsigned)	llcc68_lora_transmit()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
96	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type unsigned on 16 bits) is assigned to an object with a narrower essential type (unsigned on 8 bits)	llcc68_lora_transmit()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
241	10.1	Operands shall not be of an inappropriate essential type. The right operand of the & operator is of an inappropriate essential type category signed.	llcc68_lora_transmit()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
118	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type unsigned on 16 bits) is assigned to an object with a narrower essential type (unsigned on 8 bits)	llcc68_lora_transmit()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the

						safety of the operation.
91	10.1	Operands shall not be of an inappropriate essential type. The right operand of the & operator is of an inappropriate essential type category signed.	llcc68_lora_transmit()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
97	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type category enum) is assigned to an object with a different essential type category (unsigned)	llcc68_lora_transmit()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
157	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type category enum) is assigned to an object with a different essential type category (unsigned)	llcc68_lora_transmit()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
98	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type category enum) is assigned to an object with a different essential type category (unsigned)	llcc68_lora_transmit()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
207	10.1	Operands shall not be of an inappropriate essential type. The right operand of the = operator is of an inappropriate essential type category signed.	llcc68_lora_transmit()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
238	10.4	Both operands of an operator in which the usual arithmetic conversions are performed shall have the same essential type category. The left operand of the = operator has essentially unsigned type while the right operand has essentially signed type.	llcc68_lora_transmit()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
193	10.1	Operands shall not be of an inappropriate essential type. The left operand of the << operator is of an inappropriate essential type category signed.	llcc68_lora_transmit()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
100	10.1	Operands shall not be of an inappropriate essential type.	llcc68_lora_transmit()	Low	Not a	Embedded drivers need

		The right operand of the &= operator is of an inappropriate essential type category signed.			defect	this method to set or clear some bits and drivers guarantee the safety of the operation.
101	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type category signed) is assigned to an object with a different essential type category (unsigned)	llcc68_lora_transmit()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
197	10.4	Both operands of an operator in which the usual arithmetic conversions are performed shall have the same essential type category. The left operand of the &= operator has essentially unsigned type while the right operand has essentially signed type.	llcc68_lora_transmit()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
120	10.1	Operands shall not be of an inappropriate essential type. The operand of the ~ operator is of an inappropriate essential type category signed.	llcc68_lora_transmit()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
206	10.1	Operands shall not be of an inappropriate essential type. The left operand of the << operator is of an inappropriate essential type category signed.	llcc68_lora_transmit()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
109	10.8	The value of a composite expression shall not be cast to a different essential type category or a wider essential type. The value of the composite expression of essential type category floating shall not be cast to the different essential type category unsigned.	llcc68_lora_transmit()	Low	Not a defect	We use this function to convert driver data and drivers guarantee the safety of the operation.
104	D1.1	Any implementation-defined behaviour on which the output of the program depends shall be documented and understood. Conversion of integer to floating-point number uses an implementation-defined direction of rounding in some cases.	llcc68_lora_transmit()	Low	Justified	We use this function to convert driver settings. Developers need to refer to the data manual to convert settings.
236	10.4	Both operands of an operator in which the usual arithmetic conversions are performed shall have the same essential type category. The left operand of the / operator has essentially unsigned type	llcc68_lora_transmit()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the

		while the right operand has essentially floating type.				safety of the operation.
189	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type unsigned on 32 bits) is assigned to an object with a narrower essential type (unsigned on 8 bits)	llcc68_lora_transmit()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
110	10.1	Operands shall not be of an inappropriate essential type. The right operand of the & operator is of an inappropriate essential type category signed.	llcc68_lora_transmit()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
188	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type unsigned on 32 bits) is assigned to an object with a narrower essential type (unsigned on 8 bits)	llcc68_lora_transmit()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
79	10.1	Operands shall not be of an inappropriate essential type. The right operand of the & operator is of an inappropriate essential type category signed.	llcc68_lora_transmit()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
74	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type unsigned on 32 bits) is assigned to an object with a narrower essential type (unsigned on 8 bits)	llcc68_lora_transmit()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
130	10.1	Operands shall not be of an inappropriate essential type. The right operand of the & operator is of an inappropriate essential type category signed.	llcc68_lora_transmit()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
76	10.8	The value of a composite expression shall not be cast to a different essential type category or a wider essential type. The value of the composite expression of essential type category signed shall not be cast to the different essential type category unsigned.	llcc68_set_sleep()	Low	Not a defect	We use this function to convert driver data and drivers guarantee the safety of the operation.
249	10.1	Operands shall not be of an inappropriate essential type.	llcc68_set_sleep()	Low	Not a	Embedded drivers need

		The left operand of the << operator is of an inappropriate essential type category enum.			defect	this method to set or clear some bits and drivers guarantee the safety of the operation.
102	10.4	Both operands of an operator in which the usual arithmetic conversions are performed shall have the same essential type category. Operands of the operator have different essentially enum types.	llcc68_set_sleep()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
179	10.1	Operands shall not be of an inappropriate essential type. The left operand of the operator is of an inappropriate essential type category enum. The right operand of the operator is of an inappropriate essential type category enum.	llcc68_set_sleep()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
93	10.1	Operands shall not be of an inappropriate essential type. The left operand of the << operator is of an inappropriate essential type category enum.	llcc68_set_sleep()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
107	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type category enum) is assigned to an object with a different essential type category (unsigned)	llcc68_set_standby()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
656	D4.14	The validity of values received from external sources shall be checked. Dereferenced pointer is from an unsecure source. Pointer may be NULL or may point to unknown memory.	llcc68_set_frequency_synthesis()	Low	Justified	(handle == NULL)checked.
657	D4.14	The validity of values received from external sources shall be checked. Dereferenced pointer is from an unsecure source. Pointer may be NULL or may point to unknown memory.	llcc68_set_tx()	Low	Justified	(handle == NULL)checked.
192	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type unsigned on 32 bits) is assigned to an object with a narrower essential type (unsigned on 8 bits)	llcc68_set_tx()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.

37	10.1	Operands shall not be of an inappropriate essential type. The right operand of the & operator is of an inappropriate essential type category signed.	llcc68_set_tx()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
116	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type unsigned on 32 bits) is assigned to an object with a narrower essential type (unsigned on 8 bits)	llcc68_set_tx()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
114	10.1	Operands shall not be of an inappropriate essential type. The right operand of the & operator is of an inappropriate essential type category signed.	llcc68_set_tx()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
113	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type unsigned on 32 bits) is assigned to an object with a narrower essential type (unsigned on 8 bits)	llcc68_set_tx()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
122	10.1	Operands shall not be of an inappropriate essential type. The right operand of the & operator is of an inappropriate essential type category signed.	llcc68_set_tx()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
658	D4.14	The validity of values received from external sources shall be checked. Dereferenced pointer is from an unsecure source. Pointer may be NULL or may point to unknown memory.	llcc68_set_rx()	Low	Justified	(handle == NULL)checked.
123	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type unsigned on 32 bits) is assigned to an object with a narrower essential type (unsigned on 8 bits)	llcc68_set_rx()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
143	10.1	Operands shall not be of an inappropriate essential type. The right operand of the & operator is of an inappropriate essential type category signed.	llcc68_set_rx()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and

						drivers guarantee the safety of the operation.
71	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type unsigned on 32 bits) is assigned to an object with a narrower essential type (unsigned on 8 bits)	llcc68_set_rx()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
148	10.1	Operands shall not be of an inappropriate essential type. The right operand of the & operator is of an inappropriate essential type category signed.	llcc68_set_rx()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
127	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type unsigned on 32 bits) is assigned to an object with a narrower essential type (unsigned on 8 bits)	llcc68_set_rx()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
124	10.1	Operands shall not be of an inappropriate essential type. The right operand of the & operator is of an inappropriate essential type category signed.	llcc68_set_rx()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
180	10.8	The value of a composite expression shall not be cast to a different essential type category or a wider essential type. The value of the composite expression of essential type category floating shall not be cast to the different essential type category unsigned.	llcc68_timeout_convert_to_register()	Low	Not a defect	We use this function to convert driver data and drivers guarantee the safety of the operation.
250	D1.1	Any implementation-defined behaviour on which the output of the program depends shall be documented and understood. Conversion of integer to floating-point number uses an implementation-defined direction of rounding in some cases.	llcc68_timeout_convert_to_data()	Low	Justified	We use this function to convert driver settings. Developers need to refer to the data manual to convert settings.
160	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type category enum) is assigned to an object with a different essential type category (unsigned)	llcc68_set_stop_timer_on_preamble()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.

659	D4.14	The validity of values received from external sources shall be checked. Dereferenced pointer is from an unsecure source. Pointer may be NULL or may point to unknown memory.	llcc68_set_rx_duty_cycle()	Low	Justified	(handle == NULL)checked.
48	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type unsigned on 32 bits) is assigned to an object with a narrower essential type (unsigned on 8 bits)	llcc68_set_rx_duty_cycle()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
72	10.1	Operands shall not be of an inappropriate essential type. The right operand of the & operator is of an inappropriate essential type category signed.	llcc68_set_rx_duty_cycle()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
125	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type unsigned on 32 bits) is assigned to an object with a narrower essential type (unsigned on 8 bits)	llcc68_set_rx_duty_cycle()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
253	10.1	Operands shall not be of an inappropriate essential type. The right operand of the & operator is of an inappropriate essential type category signed.	llcc68_set_rx_duty_cycle()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
209	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type unsigned on 32 bits) is assigned to an object with a narrower essential type (unsigned on 8 bits)	llcc68_set_rx_duty_cycle()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
33	10.1	Operands shall not be of an inappropriate essential type. The right operand of the & operator is of an inappropriate essential type category signed.	llcc68_set_rx_duty_cycle()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
187	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type unsigned on 32 bits) is assigned	llcc68_set_rx_duty_cycle()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and

		to an object with a narrower essential type (unsigned on 8 bits)				drivers guarantee the safety of the operation.
217	10.1	Operands shall not be of an inappropriate essential type. The right operand of the & operator is of an inappropriate essential type category signed.	llcc68_set_rx_duty_cycle()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
156	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type unsigned on 32 bits) is assigned to an object with a narrower essential type (unsigned on 8 bits)	llcc68_set_rx_duty_cycle()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
231	10.1	Operands shall not be of an inappropriate essential type. The right operand of the & operator is of an inappropriate essential type category signed.	llcc68_set_rx_duty_cycle()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
214	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type unsigned on 32 bits) is assigned to an object with a narrower essential type (unsigned on 8 bits)	llcc68_set_rx_duty_cycle()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
62	10.1	Operands shall not be of an inappropriate essential type. The right operand of the & operator is of an inappropriate essential type category signed.	llcc68_set_rx_duty_cycle()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
660	D4.14	The validity of values received from external sources shall be checked. Dereferenced pointer is from an unsecure source. Pointer may be NULL or may point to unknown memory.	llcc68_set_cad()	Low	Justified	(handle == NULL)checked.
661	D4.14	The validity of values received from external sources shall be checked. Dereferenced pointer is from an unsecure source. Pointer may be NULL or may point to unknown memory.	llcc68_set_tx_continuous_wave()	Low	Justified	(handle == NULL)checked.
662	D4.14	The validity of values received from external sources shall be checked.	llcc68_set_tx_infinite_preamble()	Low	Justified	(handle == NULL)checked.

		Dereferenced pointer is from an unsecure source. Pointer may be NULL or may point to unknown memory.				
158	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type category enum) is assigned to an object with a different essential type category (unsigned)	llcc68_set_regulator_mode()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
663	D4.14	The validity of values received from external sources shall be checked. Dereferenced pointer is from an unsecure source. Pointer may be NULL or may point to unknown memory.	llcc68_set_calibration()	Low	Justified	(handle == NULL)checked.
664	D4.14	The validity of values received from external sources shall be checked. Dereferenced pointer is from an unsecure source. Pointer may be NULL or may point to unknown memory.	llcc68_set_calibration_image()	Low	Justified	(handle == NULL)checked.
78	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type category enum) is assigned to an object with a different essential type category (unsigned)	llcc68_set_rx_tx_fallback_mode()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
136	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type unsigned on 16 bits) is assigned to an object with a narrower essential type (unsigned on 8 bits)	llcc68_set_dio_irq_params()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
135	10.1	Operands shall not be of an inappropriate essential type. The right operand of the & operator is of an inappropriate essential type category signed.	llcc68_set_dio_irq_params()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
137	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type unsigned on 16 bits) is assigned to an object with a narrower essential type (unsigned on 8 bits)	llcc68_set_dio_irq_params()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
173	10.1	Operands shall not be of an inappropriate essential type. The right operand of the & operator is of an inappropriate essential	llcc68_set_dio_irq_params()	Low	Not a defect	Embedded drivers need this method to set or

		type category signed.				clear some bits and drivers guarantee the safety of the operation.
144	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type unsigned on 16 bits) is assigned to an object with a narrower essential type (unsigned on 8 bits)	llcc68_set_dio_irq_params()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
23	10.1	Operands shall not be of an inappropriate essential type. The right operand of the & operator is of an inappropriate essential type category signed.	llcc68_set_dio_irq_params()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
65	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type unsigned on 16 bits) is assigned to an object with a narrower essential type (unsigned on 8 bits)	llcc68_set_dio_irq_params()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
134	10.1	Operands shall not be of an inappropriate essential type. The right operand of the & operator is of an inappropriate essential type category signed.	llcc68_set_dio_irq_params()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
153	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type unsigned on 16 bits) is assigned to an object with a narrower essential type (unsigned on 8 bits)	llcc68_set_dio_irq_params()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
170	10.1	Operands shall not be of an inappropriate essential type. The right operand of the & operator is of an inappropriate essential type category signed.	llcc68_set_dio_irq_params()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
146	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type unsigned on 16 bits) is assigned to an object with a narrower essential type (unsigned on 8 bits)	llcc68_set_dio_irq_params()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the

						safety of the operation.
138	10.1	Operands shall not be of an inappropriate essential type. The right operand of the & operator is of an inappropriate essential type category signed.	llcc68_set_dio_irq_params()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
234	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type unsigned on 16 bits) is assigned to an object with a narrower essential type (unsigned on 8 bits)	llcc68_set_dio_irq_params()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
18	10.1	Operands shall not be of an inappropriate essential type. The right operand of the & operator is of an inappropriate essential type category signed.	llcc68_set_dio_irq_params()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
168	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type unsigned on 16 bits) is assigned to an object with a narrower essential type (unsigned on 8 bits)	llcc68_set_dio_irq_params()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
129	10.1	Operands shall not be of an inappropriate essential type. The right operand of the & operator is of an inappropriate essential type category signed.	llcc68_set_dio_irq_params()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
665	D4.14	The validity of values received from external sources shall be checked. Dereferenced pointer is from an unsecure source. Pointer may be NULL or may point to unknown memory.	llcc68_get_irq_status()	Low	Justified	(handle == NULL)checked.
666	D4.14	The validity of values received from external sources shall be checked. Dereferenced pointer is from an unsecure source. Pointer may be NULL or may point to unknown memory.	llcc68_get_irq_status()	Low	Justified	(handle == NULL)checked.
172	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type unsigned on 16 bits) is assigned	llcc68_clear_irq_status()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and

		to an object with a narrower essential type (unsigned on 8 bits)				drivers guarantee the safety of the operation.
32	10.1	Operands shall not be of an inappropriate essential type. The right operand of the & operator is of an inappropriate essential type category signed.	llcc68_clear_irq_status()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
203	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type unsigned on 16 bits) is assigned to an object with a narrower essential type (unsigned on 8 bits)	llcc68_clear_irq_status()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
257	10.1	Operands shall not be of an inappropriate essential type. The right operand of the & operator is of an inappropriate essential type category signed.	llcc68_clear_irq_status()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
667	D4.14	The validity of values received from external sources shall be checked. Dereferenced pointer is from an unsecure source. Pointer may be NULL or may point to unknown memory.	llcc68_set_dio2_as_rf_switch_ctrl()	Low	Justified	(handle == NULL)checked.
131	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type category enum) is assigned to an object with a different essential type category (unsigned)	llcc68_set_dio2_as_rf_switch_ctrl()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
668	D4.14	The validity of values received from external sources shall be checked. Dereferenced pointer is from an unsecure source. Pointer may be NULL or may point to unknown memory.	llcc68_set_dio3_as_tcxo_ctrl()	Low	Justified	(handle == NULL)checked.
226	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type category enum) is assigned to an object with a different essential type category (unsigned)	llcc68_set_dio3_as_tcxo_ctrl()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
166	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category.	llcc68_set_dio3_as_tcxo_ctrl()	Low	Not a defect	Embedded drivers need this method to set or

		The expression (of essential type unsigned on 32 bits) is assigned to an object with a narrower essential type (unsigned on 8 bits)				clear some bits and drivers guarantee the safety of the operation.
149	10.1	Operands shall not be of an inappropriate essential type. The right operand of the & operator is of an inappropriate essential type category signed.	llcc68_set_dio3_as_tcxo_ctrl()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
112	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type unsigned on 32 bits) is assigned to an object with a narrower essential type (unsigned on 8 bits)	llcc68_set_dio3_as_tcxo_ctrl()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
199	10.1	Operands shall not be of an inappropriate essential type. The right operand of the & operator is of an inappropriate essential type category signed.	llcc68_set_dio3_as_tcxo_ctrl()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
159	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type unsigned on 32 bits) is assigned to an object with a narrower essential type (unsigned on 8 bits)	llcc68_set_dio3_as_tcxo_ctrl()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
155	10.1	Operands shall not be of an inappropriate essential type. The right operand of the & operator is of an inappropriate essential type category signed.	llcc68_set_dio3_as_tcxo_ctrl()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
89	10.8	The value of a composite expression shall not be cast to a different essential type category or a wider essential type. The value of the composite expression of essential type category floating shall not be cast to the different essential type category unsigned.	llcc68_frequency_convert_to_register()	Low	Not a defect	We use this function to convert driver data and drivers guarantee the safety of the operation.
162	10.4	Both operands of an operator in which the usual arithmetic conversions are performed shall have the same essential type category. The left operand of the * operator has essentially signed type while	llcc68_frequency_convert_to_register()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the

		the right operand has essentially floating type.				safety of the operation.
247	10.4	Both operands of an operator in which the usual arithmetic conversions are performed shall have the same essential type category. The left operand of the * operator has essentially floating type while the right operand has essentially unsigned type.	llcc68_frequency_convert_to_register()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
99	D1.1	Any implementation-defined behaviour on which the output of the program depends shall be documented and understood. Conversion of integer to floating-point number uses an implementation-defined direction of rounding in some cases.	llcc68_frequency_convert_to_register()	Low	Justified	We use this function to convert driver settings. Developers need to refer to the data manual to convert settings.
175	10.8	The value of a composite expression shall not be cast to a different essential type category or a wider essential type. The value of the composite expression of essential type category floating shall not be cast to the different essential type category unsigned.	llcc68_frequency_convert_to_data()	Low	Not a defect	We use this function to convert driver data and drivers guarantee the safety of the operation.
163	10.4	Both operands of an operator in which the usual arithmetic conversions are performed shall have the same essential type category. The left operand of the * operator has essentially signed type while the right operand has essentially floating type.	llcc68_frequency_convert_to_data()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
141	10.4	Both operands of an operator in which the usual arithmetic conversions are performed shall have the same essential type category. The left operand of the * operator has essentially floating type while the right operand has essentially unsigned type.	llcc68_frequency_convert_to_data()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
228	D1.1	Any implementation-defined behaviour on which the output of the program depends shall be documented and understood. Conversion of integer to floating-point number uses an implementation-defined direction of rounding in some cases.	llcc68_frequency_convert_to_data()	Low	Justified	We use this function to convert driver settings. Developers need to refer to the data manual to convert settings.
219	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type unsigned on 32 bits) is assigned to an object with a narrower essential type (unsigned on 8 bits)	llcc68_set_rf_frequency()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
94	10.1	Operands shall not be of an inappropriate essential type.	llcc68_set_rf_frequency()	Low	Not a	Embedded drivers need

		The right operand of the & operator is of an inappropriate essential type category signed.			defect	this method to set or clear some bits and drivers guarantee the safety of the operation.
210	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type unsigned on 32 bits) is assigned to an object with a narrower essential type (unsigned on 8 bits)	llcc68_set_rf_frequency()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
54	10.1	Operands shall not be of an inappropriate essential type. The right operand of the & operator is of an inappropriate essential type category signed.	llcc68_set_rf_frequency()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
164	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type unsigned on 32 bits) is assigned to an object with a narrower essential type (unsigned on 8 bits)	llcc68_set_rf_frequency()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
201	10.1	Operands shall not be of an inappropriate essential type. The right operand of the & operator is of an inappropriate essential type category signed.	llcc68_set_rf_frequency()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
121	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type unsigned on 32 bits) is assigned to an object with a narrower essential type (unsigned on 8 bits)	llcc68_set_rf_frequency()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
161	10.1	Operands shall not be of an inappropriate essential type. The right operand of the & operator is of an inappropriate essential type category signed.	llcc68_set_rf_frequency()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
20	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type category enum) is assigned to an	llcc68_set_packet_type()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the

		object with a different essential type category (unsigned)				safety of the operation.
169	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type category signed) is assigned to an object with a different essential type category (unsigned)	llcc68_set_tx_params()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
152	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type category enum) is assigned to an object with a different essential type category (unsigned)	llcc68_set_tx_params()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
669	D4.14	The validity of values received from external sources shall be checked. Dereferenced pointer is from an unsecure source. Pointer may be NULL or may point to unknown memory.	llcc68_set_gfsk_modulation_params()	Low	Justified	(handle == NULL)checked.
174	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type unsigned on 32 bits) is assigned to an object with a narrower essential type (unsigned on 8 bits)	llcc68_set_gfsk_modulation_params()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
194	10.1	Operands shall not be of an inappropriate essential type. The right operand of the & operator is of an inappropriate essential type category signed.	llcc68_set_gfsk_modulation_params()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
176	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type unsigned on 32 bits) is assigned to an object with a narrower essential type (unsigned on 8 bits)	llcc68_set_gfsk_modulation_params()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
165	10.1	Operands shall not be of an inappropriate essential type. The right operand of the & operator is of an inappropriate essential type category signed.	llcc68_set_gfsk_modulation_params()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
39	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category.	llcc68_set_gfsk_modulation_params()	Low	Not a defect	Embedded drivers need this method to set or

		The expression (of essential type unsigned on 32 bits) is assigned to an object with a narrower essential type (unsigned on 8 bits)				clear some bits and drivers guarantee the safety of the operation.
182	10.1	Operands shall not be of an inappropriate essential type. The right operand of the & operator is of an inappropriate essential type category signed.	llcc68_set_gfsk_modulation_params()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
186	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type category enum) is assigned to an object with a different essential type category (unsigned)	llcc68_set_gfsk_modulation_params()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
29	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type category enum) is assigned to an object with a different essential type category (unsigned)	llcc68_set_gfsk_modulation_params()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
239	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type unsigned on 32 bits) is assigned to an object with a narrower essential type (unsigned on 8 bits)	llcc68_set_gfsk_modulation_params()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
185	10.1	Operands shall not be of an inappropriate essential type. The right operand of the & operator is of an inappropriate essential type category signed.	llcc68_set_gfsk_modulation_params()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
196	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type unsigned on 32 bits) is assigned to an object with a narrower essential type (unsigned on 8 bits)	llcc68_set_gfsk_modulation_params()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
103	10.1	Operands shall not be of an inappropriate essential type. The right operand of the & operator is of an inappropriate essential type category signed.	llcc68_set_gfsk_modulation_params()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the

						safety of the operation.
191	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type unsigned on 32 bits) is assigned to an object with a narrower essential type (unsigned on 8 bits)	llcc68_set_gfsk_modulation_params()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
254	10.1	Operands shall not be of an inappropriate essential type. The right operand of the & operator is of an inappropriate essential type category signed.	llcc68_set_gfsk_modulation_params()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
117	10.8	The value of a composite expression shall not be cast to a different essential type category or a wider essential type. The value of the composite expression of essential type category floating shall not be cast to the different essential type category unsigned.	llcc68_gfsk_bit_rate_convert_to_register()	Low	Not a defect	We use this function to convert driver data and drivers guarantee the safety of the operation.
167	10.4	Both operands of an operator in which the usual arithmetic conversions are performed shall have the same essential type category. The left operand of the * operator has essentially signed type while the right operand has essentially floating type.	llcc68_gfsk_bit_rate_convert_to_register()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
133	10.4	Both operands of an operator in which the usual arithmetic conversions are performed shall have the same essential type category. The left operand of the * operator has essentially signed type while the right operand has essentially floating type.	llcc68_gfsk_bit_rate_convert_to_register()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
411	5.1	External identifiers shall be distinct. External function llcc68_gfsk_bit_rate_convert_to_data conflicts with the external identifier llcc68_gfsk_bit_rate_convert_to_register (driver_llcc68.c line 2776).	File Scope	Low	Justified	Distinct.
225	10.8	The value of a composite expression shall not be cast to a different essential type category or a wider essential type. The value of the composite expression of essential type category floating shall not be cast to the different essential type category unsigned.	llcc68_gfsk_bit_rate_convert_to_data()	Low	Not a defect	We use this function to convert driver data and drivers guarantee the safety of the operation.
111	10.4	Both operands of an operator in which the usual arithmetic conversions are performed shall have the same essential type	llcc68_gfsk_bit_rate_convert_to_data()	Low	Not a defect	Embedded drivers need this method to set or

		category. The left operand of the * operator has essentially signed type while the right operand has essentially floating type.				clear some bits and drivers guarantee the safety of the operation.
147	10.4	Both operands of an operator in which the usual arithmetic conversions are performed shall have the same essential type category. The left operand of the / operator has essentially floating type while the right operand has essentially unsigned type.	llcc68_gfsk_bit_rate_convert_to_data()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
227	D1.1	Any implementation-defined behaviour on which the output of the program depends shall be documented and understood. Conversion of integer to floating-point number uses an implementation-defined direction of rounding in some cases.	llcc68_gfsk_bit_rate_convert_to_data()	Low	Justified	We use this function to convert driver settings. Developers need to refer to the data manual to convert settings.
251	10.8	The value of a composite expression shall not be cast to a different essential type category or a wider essential type. The value of the composite expression of essential type category floating shall not be cast to the different essential type category unsigned.	llcc68_gfsk_frequency_deviation_convert_to_register()	Low	Not a defect	We use this function to convert driver data and drivers guarantee the safety of the operation.
242	10.4	Both operands of an operator in which the usual arithmetic conversions are performed shall have the same essential type category. The left operand of the * operator has essentially floating type while the right operand has essentially unsigned type.	llcc68_gfsk_frequency_deviation_convert_to_register()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
216	D1.1	Any implementation-defined behaviour on which the output of the program depends shall be documented and understood. Conversion of integer to floating-point number uses an implementation-defined direction of rounding in some cases.	llcc68_gfsk_frequency_deviation_convert_to_register()	Low	Justified	We use this function to convert driver settings. Developers need to refer to the data manual to convert settings.
45	10.4	Both operands of an operator in which the usual arithmetic conversions are performed shall have the same essential type category. The left operand of the * operator has essentially signed type while the right operand has essentially floating type.	llcc68_gfsk_frequency_deviation_convert_to_register()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
583	5.1	External identifiers shall be distinct. External function llcc68_gfsk_frequency_deviation_convert_to_data conflicts with the external identifier llcc68_gfsk_frequency_deviation_convert_to_register	File Scope	Low	Justified	Distinct.

		(driver_llcc68.c line 2830).				
105	10.8	The value of a composite expression shall not be cast to a different essential type category or a wider essential type. The value of the composite expression of essential type category floating shall not be cast to the different essential type category unsigned.	llcc68_gfsk_frequency_deviation_convert_to_data()	Low	Not a defect	We use this function to convert driver data and drivers guarantee the safety of the operation.
252	10.4	Both operands of an operator in which the usual arithmetic conversions are performed shall have the same essential type category. The left operand of the * operator has essentially signed type while the right operand has essentially floating type.	llcc68_gfsk_frequency_deviation_convert_to_data()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
87	10.4	Both operands of an operator in which the usual arithmetic conversions are performed shall have the same essential type category. The left operand of the * operator has essentially floating type while the right operand has essentially unsigned type.	llcc68_gfsk_frequency_deviation_convert_to_data()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
230	D1.1	Any implementation-defined behaviour on which the output of the program depends shall be documented and understood. Conversion of integer to floating-point number uses an implementation-defined direction of rounding in some cases.	llcc68_gfsk_frequency_deviation_convert_to_data()	Low	Justified	We use this function to convert driver settings. Developers need to refer to the data manual to convert settings.
184	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type category enum) is assigned to an object with a different essential type category (unsigned)	llcc68_set_lora_modulation_params()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
222	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type category enum) is assigned to an object with a different essential type category (unsigned)	llcc68_set_lora_modulation_params()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
220	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type category enum) is assigned to an object with a different essential type category (unsigned)	llcc68_set_lora_modulation_params()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
151	10.3	The value of an expression shall not be assigned to an object with a	llcc68_set_lora_modulation_params()	Low	Not a	Embedded drivers need

		narrower essential type or of a different essential type category. The expression (of essential type category enum) is assigned to an object with a different essential type category (unsigned)			defect	this method to set or clear some bits and drivers guarantee the safety of the operation.
670	D4.14	The validity of values received from external sources shall be checked. Dereferenced pointer is from an unsecure source. Pointer may be NULL or may point to unknown memory.	llcc68_set_gfsk_packet_params()	Low	Justified	(handle == NULL)checked.
198	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type unsigned on 16 bits) is assigned to an object with a narrower essential type (unsigned on 8 bits)	llcc68_set_gfsk_packet_params()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
67	10.1	Operands shall not be of an inappropriate essential type. The right operand of the & operator is of an inappropriate essential type category signed.	llcc68_set_gfsk_packet_params()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
44	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type unsigned on 16 bits) is assigned to an object with a narrower essential type (unsigned on 8 bits)	llcc68_set_gfsk_packet_params()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
232	10.1	Operands shall not be of an inappropriate essential type. The right operand of the & operator is of an inappropriate essential type category signed.	llcc68_set_gfsk_packet_params()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
195	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type category enum) is assigned to an object with a different essential type category (unsigned)	llcc68_set_gfsk_packet_params()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
240	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type category enum) is assigned to an object with a different essential type category (unsigned)	llcc68_set_gfsk_packet_params()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the

						safety of the operation.
246	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type category enum) is assigned to an object with a different essential type category (unsigned)	llcc68_set_gfsk_packet_params()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
119	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type category enum) is assigned to an object with a different essential type category (unsigned)	llcc68_set_gfsk_packet_params()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
171	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type category enum) is assigned to an object with a different essential type category (unsigned)	llcc68_set_gfsk_packet_params()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
77	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type unsigned on 16 bits) is assigned to an object with a narrower essential type (unsigned on 8 bits)	llcc68_set_lora_packet_params()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
205	10.1	Operands shall not be of an inappropriate essential type. The right operand of the & operator is of an inappropriate essential type category signed.	llcc68_set_lora_packet_params()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
61	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type unsigned on 16 bits) is assigned to an object with a narrower essential type (unsigned on 8 bits)	llcc68_set_lora_packet_params()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
200	10.1	Operands shall not be of an inappropriate essential type. The right operand of the & operator is of an inappropriate essential type category signed.	llcc68_set_lora_packet_params()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
21	10.3	The value of an expression shall not be assigned to an object with a	llcc68_set_lora_packet_params()	Low	Not a	Embedded drivers need

		narrower essential type or of a different essential type category. The expression (of essential type category enum) is assigned to an object with a different essential type category (unsigned)			defect	this method to set or clear some bits and drivers guarantee the safety of the operation.
211	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type category enum) is assigned to an object with a different essential type category (unsigned)	llcc68_set_lora_packet_params()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
221	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type category enum) is assigned to an object with a different essential type category (unsigned)	llcc68_set_lora_packet_params()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
57	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type category enum) is assigned to an object with a different essential type category (unsigned)	llcc68_set_cad_params()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
150	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type category enum) is assigned to an object with a different essential type category (unsigned)	llcc68_set_cad_params()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
190	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type unsigned on 32 bits) is assigned to an object with a narrower essential type (unsigned on 8 bits)	llcc68_set_cad_params()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
41	10.1	Operands shall not be of an inappropriate essential type. The right operand of the & operator is of an inappropriate essential type category signed.	llcc68_set_cad_params()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
139	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type unsigned on 32 bits) is assigned	llcc68_set_cad_params()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the

		to an object with a narrower essential type (unsigned on 8 bits)				safety of the operation.
178	10.1	Operands shall not be of an inappropriate essential type. The right operand of the & operator is of an inappropriate essential type category signed.	llcc68_set_cad_params()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
177	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type unsigned on 32 bits) is assigned to an object with a narrower essential type (unsigned on 8 bits)	llcc68_set_cad_params()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
233	10.1	Operands shall not be of an inappropriate essential type. The right operand of the & operator is of an inappropriate essential type category signed.	llcc68_set_cad_params()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
671	D4.14	The validity of values received from external sources shall be checked. Dereferenced pointer is from an unsecure source. Pointer may be NULL or may point to unknown memory.	llcc68_get_rx_buffer_status()	Low	Justified	(handle == NULL)checked.
672	D4.14	The validity of values received from external sources shall be checked. Dereferenced pointer is from an unsecure source. Pointer may be NULL or may point to unknown memory.	llcc68_get_rx_buffer_status()	Low	Justified	(handle == NULL)checked.
673	D4.14	The validity of values received from external sources shall be checked. Dereferenced pointer is from an unsecure source. Pointer may be NULL or may point to unknown memory.	llcc68_get_rx_buffer_status()	Low	Justified	(handle == NULL)checked.
674	D4.14	The validity of values received from external sources shall be checked. Dereferenced pointer is from an unsecure source. Pointer may be NULL or may point to unknown memory.	llcc68_get_gfsk_packet_status()	Low	Justified	(handle == NULL)checked.
675	D4.14	The validity of values received from external sources shall be checked. Dereferenced pointer is from an unsecure source. Pointer may be NULL or may point to unknown memory.	llcc68_get_gfsk_packet_status()	Low	Justified	(handle == NULL)checked.

676	D4.14	The validity of values received from external sources shall be checked. Dereferenced pointer is from an unsecure source. Pointer may be NULL or may point to unknown memory.	llcc68_get_gfsk_packet_status()	Low	Justified	(handle == NULL)checked.
677	D4.14	The validity of values received from external sources shall be checked. Dereferenced pointer is from an unsecure source. Pointer may be NULL or may point to unknown memory.	llcc68_get_gfsk_packet_status()	Low	Justified	(handle == NULL)checked.
678	D4.14	The validity of values received from external sources shall be checked. Dereferenced pointer is from an unsecure source. Pointer may be NULL or may point to unknown memory.	llcc68_get_gfsk_packet_status()	Low	Justified	(handle == NULL)checked.
679	D4.14	The validity of values received from external sources shall be checked. Dereferenced pointer is from an unsecure source. Pointer may be NULL or may point to unknown memory.	llcc68_get_gfsk_packet_status()	Low	Justified	(handle == NULL)checked.
691	D4.14	The validity of values received from external sources shall be checked. Dereferenced pointer is from an unsecure source. Pointer may be NULL or may point to unknown memory.	llcc68_get_lora_packet_status()	Low	Justified	(handle == NULL)checked.
692	D4.14	The validity of values received from external sources shall be checked. Dereferenced pointer is from an unsecure source. Pointer may be NULL or may point to unknown memory.	llcc68_get_lora_packet_status()	Low	Justified	(handle == NULL)checked.
680	D4.14	The validity of values received from external sources shall be checked. Dereferenced pointer is from an unsecure source. Pointer may be NULL or may point to unknown memory.	llcc68_get_instantaneous_rssi()	Low	Justified	(handle == NULL)checked.
681	D4.14	The validity of values received from external sources shall be checked. Dereferenced pointer is from an unsecure source. Pointer may be NULL or may point to unknown memory.	llcc68_get_instantaneous_rssi()	Low	Justified	(handle == NULL)checked.
682	D4.14	The validity of values received from external sources shall be checked. Dereferenced pointer is from an unsecure source. Pointer may be NULL or may point to unknown memory.	llcc68_get_instantaneous_rssi()	Low	Justified	(handle == NULL)checked.

683	D4.14	The validity of values received from external sources shall be checked. Dereferenced pointer is from an unsecure source. Pointer may be NULL or may point to unknown memory.	llcc68_get_stats()	Low	Justified	(handle == NULL)checked.
684	D4.14	The validity of values received from external sources shall be checked. Dereferenced pointer is from an unsecure source. Pointer may be NULL or may point to unknown memory.	llcc68_get_stats()	Low	Justified	(handle == NULL)checked.
685	D4.14	The validity of values received from external sources shall be checked. Dereferenced pointer is from an unsecure source. Pointer may be NULL or may point to unknown memory.	llcc68_get_stats()	Low	Justified	(handle == NULL)checked.
686	D4.14	The validity of values received from external sources shall be checked. Dereferenced pointer is from an unsecure source. Pointer may be NULL or may point to unknown memory.	llcc68_get_stats()	Low	Justified	(handle == NULL)checked.
223	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type unsigned on 16 bits) is assigned to an object with a narrower essential type (unsigned on 8 bits)	llcc68_reset_stats()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
261	10.1	Operands shall not be of an inappropriate essential type. The right operand of the & operator is of an inappropriate essential type category signed.	llcc68_reset_stats()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
47	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type unsigned on 16 bits) is assigned to an object with a narrower essential type (unsigned on 8 bits)	llcc68_reset_stats()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
224	10.1	Operands shall not be of an inappropriate essential type. The right operand of the & operator is of an inappropriate essential type category signed.	llcc68_reset_stats()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
142	10.3	The value of an expression shall not be assigned to an object with a	llcc68_reset_stats()	Low	Not a	Embedded drivers need

		narrower essential type or of a different essential type category. The expression (of essential type unsigned on 16 bits) is assigned to an object with a narrower essential type (unsigned on 8 bits)			defect	this method to set or clear some bits and drivers guarantee the safety of the operation.
235	10.1	Operands shall not be of an inappropriate essential type. The right operand of the & operator is of an inappropriate essential type category signed.	llcc68_reset_stats()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
243	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type unsigned on 16 bits) is assigned to an object with a narrower essential type (unsigned on 8 bits)	llcc68_reset_stats()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
181	10.1	Operands shall not be of an inappropriate essential type. The right operand of the & operator is of an inappropriate essential type category signed.	llcc68_reset_stats()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
259	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type unsigned on 16 bits) is assigned to an object with a narrower essential type (unsigned on 8 bits)	llcc68_reset_stats()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
215	10.1	Operands shall not be of an inappropriate essential type. The right operand of the & operator is of an inappropriate essential type category signed.	llcc68_reset_stats()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
248	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type unsigned on 16 bits) is assigned to an object with a narrower essential type (unsigned on 8 bits)	llcc68_reset_stats()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
64	10.1	Operands shall not be of an inappropriate essential type. The right operand of the & operator is of an inappropriate essential type category signed.	llcc68_reset_stats()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the

						safety of the operation.
687	D4.14	The validity of values received from external sources shall be checked. Dereferenced pointer is from an unsecure source. Pointer may be NULL or may point to unknown memory.	llcc68_get_device_errors()	Low	Justified	(handle == NULL)checked.
688	D4.14	The validity of values received from external sources shall be checked. Dereferenced pointer is from an unsecure source. Pointer may be NULL or may point to unknown memory.	llcc68_get_device_errors()	Low	Justified	(handle == NULL)checked.
128	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type unsigned on 16 bits) is assigned to an object with a narrower essential type (unsigned on 8 bits)	llcc68_set_fsk_whitening_initial_value()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
80	10.1	Operands shall not be of an inappropriate essential type. The right operand of the & operator is of an inappropriate essential type category signed.	llcc68_set_fsk_whitening_initial_value()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
202	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type unsigned on 16 bits) is assigned to an object with a narrower essential type (unsigned on 8 bits)	llcc68_set_fsk_whitening_initial_value()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
26	10.1	Operands shall not be of an inappropriate essential type. The right operand of the & operator is of an inappropriate essential type category signed.	llcc68_set_fsk_whitening_initial_value()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
53	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type unsigned on 16 bits) is assigned to an object with a narrower essential type (unsigned on 8 bits)	llcc68_set_fsk_crc_initical_value()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
66	10.1	Operands shall not be of an inappropriate essential type. The right operand of the & operator is of an inappropriate essential type category signed.	llcc68_set_fsk_crc_initical_value()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and

						drivers guarantee the safety of the operation.
132	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type unsigned on 16 bits) is assigned to an object with a narrower essential type (unsigned on 8 bits)	llcc68_set_fsk_crc_initical_value()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
28	10.1	Operands shall not be of an inappropriate essential type. The right operand of the & operator is of an inappropriate essential type category signed.	llcc68_set_fsk_crc_initical_value()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
140	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type unsigned on 16 bits) is assigned to an object with a narrower essential type (unsigned on 8 bits)	llcc68_set_fsk_crc_polynomial_value()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
208	10.1	Operands shall not be of an inappropriate essential type. The right operand of the & operator is of an inappropriate essential type category signed.	llcc68_set_fsk_crc_polynomial_value()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
256	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type unsigned on 16 bits) is assigned to an object with a narrower essential type (unsigned on 8 bits)	llcc68_set_fsk_crc_polynomial_value()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
213	10.1	Operands shall not be of an inappropriate essential type. The right operand of the & operator is of an inappropriate essential type category signed.	llcc68_set_fsk_crc_polynomial_value()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
85	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type unsigned on 16 bits) is assigned to an object with a narrower essential type (unsigned on 8 bits)	llcc68_set_lora_sync_word()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.

229	10.1	Operands shall not be of an inappropriate essential type. The right operand of the & operator is of an inappropriate essential type category signed.	llcc68_set_lora_sync_word()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
260	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type unsigned on 16 bits) is assigned to an object with a narrower essential type (unsigned on 8 bits)	llcc68_set_lora_sync_word()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
258	10.1	Operands shall not be of an inappropriate essential type. The right operand of the & operator is of an inappropriate essential type category signed.	llcc68_set_lora_sync_word()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
689	D4.14	The validity of values received from external sources shall be checked. Dereferenced pointer is from an unsecure source. Pointer may be NULL or may point to unknown memory.	llcc68_write_read_reg()	Low	Justified	(handle == NULL)checked.

Table 2.4. E:\Github\llcc68\src\driver_llcc68.h

ID	Guideline	Message	Function	Severity	Status	Comment
5	12.2	The right hand operand of a shift operator shall lie in the range zero to one less than the width in bits of the essential type of the left hand operand.	File Scope	Low	Justified	Can't be.
10	12.2	The right hand operand of a shift operator shall lie in the range zero to one less than the width in bits of the essential type of the left hand operand.	File Scope	Low	Justified	Can't be.
3	12.2	The right hand operand of a shift operator shall lie in the range zero to one less than the width in bits of the essential type of the left hand operand.	File Scope	Low	Justified	Can't be.
648	5.1	External identifiers shall be distinct. External function llcc68_gfsk_bit_rate_convert_to_data conflicts with the external identifier llcc68_gfsk_bit_rate_convert_to_register (driver_llcc68.c line 2776).	File Scope	Low	Justified	Distinct.
355	5.1	External identifiers shall be distinct.	File Scope	Low	Justified	Distinct.

		External function llcc68_gfsk_frequency_deviation_convert_to_data conflicts with the external identifier llcc68_gfsk_frequency_deviation_convert_to_register (driver_llcc68.c line 2830).				
--	--	---	--	--	--	--

Table 2.5. E:\Github\llcc68\test\driver_llcc68_cad_test.c

ID	Guideline	Message	Function	Severity	Status	Comment
616	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
443	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
367	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
360	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
554	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
398	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
399	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
519	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
552	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
363	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
401	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
524	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
393	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.

613	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
365	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
356	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
403	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
388	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
387	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
535	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
16	10.1	Operands shall not be of an inappropriate essential type. The right operand of the = operator is of an inappropriate essential type category signed.	llcc68_cad_test()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
391	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
376	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
389	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
626	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
17	10.1	Operands shall not be of an inappropriate essential type. The right operand of the = operator is of an inappropriate essential type category signed.	llcc68_cad_test()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
437	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
378	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.

377	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
533	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
570	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
371	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.

Table 2.6. E:\Github\llcc68\test\driver_llcc68_register_test.c

ID	Guideline	Message	Function	Severity	Status	Comment
641	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
620	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
447	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
615	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
628	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
416	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
651	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
596	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
618	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
548	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
405	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.

644	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
587	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
643	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
328	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type category signed) is assigned to an object with a different essential type category (unsigned)	llcc68_register_test()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
522	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
589	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
619	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
597	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
623	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
329	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type category signed) is assigned to an object with a different essential type category (unsigned)	llcc68_register_test()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
426	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
614	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
390	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
556	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
590	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.

330	10.4	Both operands of an operator in which the usual arithmetic conversions are performed shall have the same essential type category. The left operand of the / operator has essentially signed type while the right operand has essentially floating type.	llcc68_register_test()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
592	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
484	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
630	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
558	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
395	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
550	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
472	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
572	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
612	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
566	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
571	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
622	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
547	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
560	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
544	2.2	There shall be no dead code.	File Scope	Low	Justified	print function.

		The call to function llcc68_interface_debug_print has no effect.				
542	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
581	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
520	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
515	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
642	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
495	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
546	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
575	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
563	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
534	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
577	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
601	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
504	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
645	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
353	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
636	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.

632	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
410	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
553	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
343	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type category signed) is assigned to an object with a different essential type category (unsigned)	llcc68_register_test()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
549	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
467	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
562	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
586	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
478	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
336	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type category signed) is assigned to an object with a different essential type category (unsigned)	llcc68_register_test()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
531	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
350	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
639	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
555	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
351	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.

334	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type category signed) is assigned to an object with a different essential type category (unsigned)	llcc68_register_test()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
418	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
579	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
609	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
574	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
518	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
327	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type category signed) is assigned to an object with a different essential type category (unsigned)	llcc68_register_test()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
573	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
517	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
396	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
598	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
513	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
333	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type category signed) is assigned to an object with a different essential type category (unsigned)	llcc68_register_test()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
580	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.

434	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
440	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
511	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
568	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
335	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type category signed) is assigned to an object with a different essential type category (unsigned)	llcc68_register_test()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
510	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
625	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
357	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
627	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
508	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
332	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type category signed) is assigned to an object with a different essential type category (unsigned)	llcc68_register_test()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
646	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
500	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
621	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
499	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.

631	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
551	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
638	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
386	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
584	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
341	10.1	Operands shall not be of an inappropriate essential type. The right operand of the = operator is of an inappropriate essential type category signed.	llcc68_register_test()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
543	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
491	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
505	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
346	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
497	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
344	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type category signed) is assigned to an object with a different essential type category (unsigned)	llcc68_register_test()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
404	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
569	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
383	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.

607	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
493	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
331	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type category signed) is assigned to an object with a different essential type category (unsigned)	llcc68_register_test()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
436	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
594	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
567	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
603	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
611	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
338	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type category signed) is assigned to an object with a different essential type category (unsigned)	llcc68_register_test()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
536	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
496	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
650	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
576	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
487	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
485	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.

490	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
593	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
461	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
492	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
523	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
529	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
617	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
413	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
525	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
606	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
373	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
471	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
652	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
600	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
608	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
483	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
604	2.2	There shall be no dead code.	File Scope	Low	Justified	print function.

		The call to function llcc68_interface_debug_print has no effect.				
488	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
417	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
480	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
456	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
474	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
489	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
479	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
339	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type category signed) is assigned to an object with a different essential type category (unsigned)	llcc68_register_test()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
516	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
361	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
473	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
503	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
635	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
591	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
340	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type category signed) is assigned to an	llcc68_register_test()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee

		object with a different essential type category (unsigned)				the safety of the operation.
476	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
475	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
501	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
509	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
532	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
342	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type category signed) is assigned to an object with a different essential type category (unsigned)	llcc68_register_test()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
599	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
382	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
424	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
444	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
521	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
582	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
337	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type category signed) is assigned to an object with a different essential type category (unsigned)	llcc68_register_test()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
432	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.

498	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
400	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
528	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
605	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.

Table 2.7. E:\Github\llcc68\test\driver_llcc68_sent_receive_test.c

ID	Guideline	Message	Function	Severity	Status	Comment
316	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type category enum) is assigned to an object with a different essential type category (unsigned)	a_callback()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
459	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
311	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type category enum) is assigned to an object with a different essential type category (unsigned)	a_callback()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
647	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
512	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
450	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
394	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
308	10.1	Operands shall not be of an inappropriate essential type. The right operand of the && operator is of an inappropriate essential type category unsigned.	a_callback()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
494	2.2	There shall be no dead code.	File Scope	Low	Justified	print function.

		The call to function llcc68_interface_debug_print has no effect.				
352	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
324	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type category enum) is assigned to an object with a different essential type category (unsigned)	a_callback()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
454	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
314	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type category enum) is assigned to an object with a different essential type category (unsigned)	a_callback()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
464	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
312	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type category enum) is assigned to an object with a different essential type category (unsigned)	a_callback()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
539	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
313	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type category enum) is assigned to an object with a different essential type category (unsigned)	a_callback()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
380	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
315	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type category enum) is assigned to an object with a different essential type category (unsigned)	a_callback()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
649	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
310	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category.	a_callback()	Low	Not a defect	Embedded drivers need this method to set or clear some

		The expression (of essential type category enum) is assigned to an object with a different essential type category (unsigned)				bits and drivers guarantee the safety of the operation.
506	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
317	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type category enum) is assigned to an object with a different essential type category (unsigned)	a_callback()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
439	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
322	10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category. The expression (of essential type category enum) is assigned to an object with a different essential type category (unsigned)	a_callback()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
468	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
449	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
381	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
446	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
466	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
420	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
640	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
414	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
469	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
462	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.

348	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
429	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
465	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
408	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
451	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
526	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
477	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
537	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
445	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
392	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
585	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
319	10.1	Operands shall not be of an inappropriate essential type. The right operand of the = operator is of an inappropriate essential type category signed.	llcc68_sent_test()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
458	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
538	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
514	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
442	2.2	There shall be no dead code.	File Scope	Low	Justified	print function.

		The call to function llcc68_interface_debug_print has no effect.				
309	10.1	Operands shall not be of an inappropriate essential type. The right operand of the = operator is of an inappropriate essential type category signed.	llcc68_sent_test()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
370	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
320	10.4	Both operands of an operator in which the usual arithmetic conversions are performed shall have the same essential type category. The left operand of the < operator has essentially unsigned type while the right operand has essentially signed type.	llcc68_sent_test()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
561	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
565	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
557	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
452	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
441	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
430	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
397	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
633	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
564	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
457	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
578	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.

624	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
541	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
409	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
455	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
448	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
435	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
369	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
385	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
486	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
588	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
375	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
323	10.1	Operands shall not be of an inappropriate essential type. The right operand of the = operator is of an inappropriate essential type category signed.	llcc68_receive_test()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
433	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
425	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
481	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
460	2.2	There shall be no dead code.	File Scope	Low	Justified	print function.

		The call to function llcc68_interface_debug_print has no effect.				
318	10.1	Operands shall not be of an inappropriate essential type. The right operand of the = operator is of an inappropriate essential type category signed.	llcc68_receive_test()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
415	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
431	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
527	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
325	10.1	Operands shall not be of an inappropriate essential type. The right operand of the = operator is of an inappropriate essential type category signed.	llcc68_receive_test()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
326	10.4	Both operands of an operator in which the usual arithmetic conversions are performed shall have the same essential type category. The left operand of the = operator has essentially unsigned type while the right operand has essentially signed type.	llcc68_receive_test()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
321	10.1	Operands shall not be of an inappropriate essential type. The left operand of the << operator is of an inappropriate essential type category signed.	llcc68_receive_test()	Low	Not a defect	Embedded drivers need this method to set or clear some bits and drivers guarantee the safety of the operation.
602	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
372	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
470	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
653	D4.14	The validity of values received from external sources shall be checked. Loop is controlled by a value from an unsecure source. Loop may be infinite.	llcc68_receive_test()	Low	Justified	Loop can't be infinite.
540	2.2	There shall be no dead code. The call to function llcc68_interface_delay_ms has no effect.	File Scope	Low	Justified	delay function.

654	14.3	Controlling expressions shall not be invariant. If condition is always true.	llcc68_receive_test()	Low	Justified	Can't be.
463	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.
655	2.1	A project shall not contain unreachable code. The else branch is unreachable.	llcc68_receive_test()	Low	Justified	Can't be.
384	2.2	There shall be no dead code. The call to function llcc68_interface_debug_print has no effect.	File Scope	Low	Justified	print function.

Chapter 3. Defects

Defects

No defects were found.

Chapter 4. Appendix 1 - Configuration Settings

Polyspace Settings

Option	Value
-author	LibDriver
-bug-finder	true
-checkers	ALIGNMENT_CHANGE, ASSERT, ATOMIC_VAR_ACCESS_TWICE, ATOMIC_VAR_SEQUENCE_NOT_ATOMIC, BAD_EQUAL_EQUAL_USE, BAD_EQUAL_USE, BAD_FREE, BAD_LOCK, BAD_PTR_SCALING, BAD_UNLOCK, CHARACTER_MISUSE, CHAR_EOF_CONFUSED, CLOSED_RESOURCE_USE, CONSTANT_OBJECT_WRITE, DATA_RACE, DATA_RACE_STD_LIB, DEADLOCK, DECL_MISMATCH, DOUBLE_DEALLOCATION, DOUBLE_LOCK, DOUBLE_RESOURCE_CLOSE, DOUBLE_RESOURCE_OPEN, DOUBLE_UNLOCK, ERRNO_MISUSE, FILE_OBJECT_MISUSE, FLEXIBLE_ARRAY_MEMBER_STRUCT_MISUSE, FLOAT_ABSORPTION, FLOAT_CONV_OVFL, FLOAT_STD_LIB, FLOAT_ZERO_DIV, FREED_PTR, FUNC_CAST, IMPROPER_ARRAY_INIT, INLINE_CONSTRAINT_NOT_RESPECTED, INT_CONV_OVFL, INT_STD_LIB, INT_ZERO_DIV, INVALID_ENV_POINTER, INVALID_MEMORY_ASSUMPTION, INVALID_VA_LIST_ARG, IO_INTERLEAVING, LOCAL_ADDR_ESCAPE, MACRO_USED_AS_OBJECT, MEMCMP_PADDING_DATA, MEMCMP_STRINGS, MEM_STD_LIB, MISSING_ERRNO_RESET, MISSING_NULL_CHAR, MISSING_RETURN, NON_INIT_PTR, NON_INIT_VAR, NON_POSITIVE_VLA_SIZE, NULL_PTR, OPERATOR_PRECEDENCE, OTHER_STD_LIB, OUT_BOUND_ARRAY, OUT_BOUND_PTR, PARTIALLY_ACCESSED_ARRAY, PRE_DIRECTIVE_MACRO_ARG, PRE_UCNAME_JOIN_TOKENS, PTR_CAST, PTR_SIZEOF_MISMATCH, PTR_TO_DIFF_ARRAY, PUTENV_AUTO_VAR, READ_ONLY_RESOURCE_WRITE, RESOURCE_LEAK, SIDE_EFFECT_IGNORED, SIGN_CHANGE, SIG_HANDLER_CALLING_SIGNAL, SIG_HANDLER_COMP_EXCP_RETURN, SIG_HANDLER_ERRNO_MISUSE, SIG_HANDLER_SHARED_OBJECT, SIZEOF_MISUSE, STD_FUNC_ARG_MISMATCH, STREAM_WITH_SIDE_EFFECT, STRING_FORMAT, STRLIB_BUFFER_OVERFLOW, STRLIB_BUFFER_UNDERFLOW, STR_FORMAT_BUFFER_OVERFLOW, STR_STD_LIB, TEMP_OBJECT_ACCESS, TOO_MANY_VA_ARG_CALLS, TYPEDEF_MISMATCH, UINT_CONV_OVFL, UNPROTOTYPED_FUNC_CALL, UNREACHABLE, USELESS_IF, USELESS_WRITE, VAR_SHADOWING, VA_ARG_INCORRECT_TYPE, VA_START_INCORRECT_TYPE, VA_START_MISUSE
-compiler	iar
-D	__TID__=14, __SIZE_T_TYPE__=unsigned int, __PTRDIFF_T_TYPE__=signed int, __IAR_SYSTEMS_ICC=1
-date	02/04/2023
-dos	true
-I	E:\Github\lcc68\src,E:\Github\lcc68\interface,E:\Github\lcc68\example,E:\Github\lcc68\test
-import-comments	E:\Polyspace\lcc68\Module\BF_Result\comments_bak
-lang	C

-little-endian	true
-logical-signed-right-shift	true
-misra3	mandatory-required
-prog	llcc68
-results-dir	E:\Polyspace\llcc68\Module\BF_Result
-sfr-types	sfr8=8,sfr16=16,sfr32=32,sfr=8
-target	mcpu
-verif-version	1.0

Coding Standard Configuration

Table 4.1. MISRA C:2012 Guidelines Configuration

Guideline	Description	Mode	Comment	Enabled
D1.1	Any implementation-defined behaviour on which the output of the program depends shall be documented and understood.	required	-	yes
D2.1	All source files shall compile without any compilation errors.	required	-	yes
D3.1	All code shall be traceable to documented requirements.	required	Not enforceable	no
D4.1	Run-time failures shall be minimized.	required	-	yes
D4.2	All usage of assembly language should be documented.	advisory	Not enforceable	no
D4.3	Assembly language shall be encapsulated and isolated.	required	-	yes
D4.4	Sections of code should not be "commented out".	advisory	Not implemented	no
D4.5	Identifiers in the same name space with overlapping visibility should be typographically unambiguous.	advisory	-	no
D4.6	typedefs that indicate size and signedness should be used in place of the basic numerical types.	advisory	-	no
D4.7	If a function returns error information, then that error information shall be tested.	required	-	yes
D4.8	If a pointer to a structure or union is never dereferenced within a translation unit, then the implementation of the object should be hidden.	advisory	-	no
D4.9	A function should be used in preference to a function-like macro where they are interchangeable.	advisory	-	no
D4.10	Precautions shall be taken in order to prevent the contents of a header file being included more than once.	required	-	yes

D4.11	The validity of values passed to library functions shall be checked.	required	-	yes
D4.12	Dynamic memory allocation shall not be used.	required	-	yes
D4.13	Functions which are designed to provide operations on a resource should be called in an appropriate sequence.	advisory	-	no
D4.14	The validity of values received from external sources shall be checked.	required	-	yes
1.1	The program shall contain no violations of the standard C syntax and constraints, and shall not exceed the implementation's translation limits.	required	-	yes
1.2	Language extensions should not be used.	advisory	-	no
1.3	There shall be no occurrence of undefined or critical unspecified behaviour.	required	-	yes
2.1	A project shall not contain unreachable code.	required	-	yes
2.2	There shall be no dead code.	required	-	yes
2.3	A project should not contain unused type declarations.	advisory	-	no
2.4	A project should not contain unused tag declarations.	advisory	-	no
2.5	A project should not contain unused macro declarations.	advisory	-	no
2.6	A function should not contain unused label declarations.	advisory	-	no
2.7	There should be no unused parameters in functions.	advisory	-	no
3.1	The character sequences /* and // shall not be used within a comment.	required	-	yes
3.2	Line-splicing shall not be used in // comments.	required	-	yes
4.1	Octal and hexadecimal escape sequences shall be terminated.	required	-	yes
4.2	Trigraphs should not be used.	advisory	-	no
5.1	External identifiers shall be distinct.	required	-	yes
5.2	Identifiers declared in the same scope and name space shall be distinct.	required	-	yes
5.3	An identifier declared in an inner scope shall not hide an identifier declared in an outer scope.	required	-	yes
5.4	Macro identifiers shall be distinct.	required	-	yes
5.5	Identifiers shall be distinct from macro names.	required	-	yes
5.6	A typedef name shall be a unique identifier.	required	-	yes
5.7	A tag name shall be a unique identifier.	required	-	yes
5.8	Identifiers that define objects or functions with external linkage shall be unique.	required	-	yes

5.9	Identifiers that define objects or functions with internal linkage should be unique.	advisory	-	no
6.1	Bit-fields shall only be declared with an appropriate type.	required	-	yes
6.2	Single-bit named bit fields shall not be of a signed type.	required	-	yes
7.1	Octal constants shall not be used.	required	-	yes
7.2	A "u" or "U" suffix shall be applied to all integer constants that are represented in an unsigned type.	required	-	yes
7.3	The lowercase character "l" shall not be used in a literal suffix.	required	-	yes
7.4	A string literal shall not be assigned to an object unless the object's type is "pointer to const-qualified char".	required	-	yes
8.1	Types shall be explicitly specified.	required	-	yes
8.2	Function types shall be in prototype form with named parameters.	required	-	yes
8.3	All declarations of an object or function shall use the same names and type qualifiers.	required	-	yes
8.4	A compatible declaration shall be visible when an object or function with external linkage is defined.	required	-	yes
8.5	An external object or function shall be declared once in one and only one file.	required	-	yes
8.6	An identifier with external linkage shall have exactly one external definition.	required	-	yes
8.7	Functions and objects should not be defined with external linkage if they are referenced in only one translation unit.	advisory	-	no
8.8	The static storage class specifier shall be used in all declarations of objects and functions that have internal linkage.	required	-	yes
8.9	An object should be defined at block scope if its identifier only appears in a single function.	advisory	-	no
8.10	An inline function shall be declared with the static storage class.	required	-	yes
8.11	When an array with external linkage is declared, its size should be explicitly specified.	advisory	-	no
8.12	Within an enumerator list, the value of an implicitly-specified enumeration constant shall be unique.	required	-	yes
8.13	A pointer should point to a const-qualified type whenever possible.	advisory	-	no
8.14	The restrict type qualifier shall not be used.	required	-	yes
9.1	The value of an object with automatic storage duration shall not be read before it has been set.	mandatory	-	yes
9.2	The initializer for an aggregate or union shall be enclosed in braces.	required	-	yes
9.3	Arrays shall not be partially initialized.	required	-	yes
9.4	An element of an object shall not be initialized more than once.	required	-	yes

9.5	Where designated initializers are used to initialize an array object the size of the array shall be specified explicitly.	required	-	yes
10.1	Operands shall not be of an inappropriate essential type.	required	-	yes
10.2	Expressions of essentially character type shall not be used inappropriately in addition and subtraction operations.	required	-	yes
10.3	The value of an expression shall not be assigned to an object with a narrower essential type or of a different essential type category.	required	-	yes
10.4	Both operands of an operator in which the usual arithmetic conversions are performed shall have the same essential type category.	required	-	yes
10.5	The value of an expression should not be cast to an inappropriate essential type.	advisory	-	no
10.6	The value of a composite expression shall not be assigned to an object with wider essential type.	required	-	yes
10.7	If a composite expression is used as one operand of an operator in which the usual arithmetic conversions are performed then the other operand shall not have wider essential type.	required	-	yes
10.8	The value of a composite expression shall not be cast to a different essential type category or a wider essential type.	required	-	yes
11.1	Conversions shall not be performed between a pointer to a function and any other type.	required	-	yes
11.2	Conversions shall not be performed between a pointer to an incomplete type and any other type.	required	-	yes
11.3	A cast shall not be performed between a pointer to object type and a pointer to a different object type.	required	-	yes
11.4	A conversion should not be performed between a pointer to object and an integer type.	advisory	-	no
11.5	A conversion should not be performed from pointer to void into pointer to object.	advisory	-	no
11.6	A cast shall not be performed between pointer to void and an arithmetic type.	required	-	yes
11.7	A cast shall not be performed between pointer to object and a non-integer arithmetic type.	required	-	yes
11.8	A cast shall not remove any const or volatile qualification from the type pointed to by a pointer.	required	-	yes
11.9	The macro NULL shall be the only permitted form of integer null pointer constant.	required	-	yes
12.1	The precedence of operators within expressions should be made explicit.	advisory	-	no
12.2	The right hand operand of a shift operator shall lie in the range zero to one less than the width in bits of the essential type of the left hand operand.	required	-	yes
12.3	The comma operator should not be used	advisory	-	no
12.4	Evaluation of constant expressions should not lead to unsigned integer wrap-around.	advisory	-	no
12.5	The sizeof operator shall not have an operand which is a function parameter declared as "array of	mandatory	-	yes

	type".			
13.1	Initializer lists shall not contain persistent side effects.	required	-	yes
13.2	The value of an expression and its persistent side effects shall be the same under all permitted evaluation orders.	required	-	yes
13.3	A full expression containing an increment (++) or decrement (--) operator should have no other potential side effects other than that caused by the increment or decrement operator.	advisory	-	no
13.4	The result of an assignment operator should not be used.	advisory	-	no
13.5	The right hand operand of a logical && or operator shall not contain persistent side effects.	required	-	yes
13.6	The operand of the sizeof operator shall not contain any expression which has potential side effects.	mandatory	-	yes
14.1	A loop counter shall not have essentially floating type.	required	-	yes
14.2	A for loop shall be well-formed.	required	-	yes
14.3	Controlling expressions shall not be invariant.	required	-	yes
14.4	The controlling expression of an if statement and the controlling expression of an iteration-statement shall have essentially Boolean type.	required	-	yes
15.1	The goto statement should not be used.	advisory	-	no
15.2	The goto statement shall jump to a label declared later in the same function.	required	-	yes
15.3	Any label referenced by a goto statement shall be declared in the same block, or in any block enclosing the goto statement.	required	-	yes
15.4	There should be no more than one break or goto statement used to terminate any iteration statement.	advisory	-	no
15.5	A function should have a single point of exit at the end.	advisory	-	no
15.6	The body of an iteration-statement or a selection-statement shall be a compound-statement.	required	-	yes
15.7	All if ... else if constructs shall be terminated with an else statement.	required	-	yes
16.1	All switch statements shall be well-formed.	required	-	yes
16.2	A switch label shall only be used when the most closely-enclosing compound statement is the body of a switch statement.	required	-	yes
16.3	An unconditional break statement shall terminate every switch-clause.	required	-	yes
16.4	Every switch statement shall have a default label.	required	-	yes
16.5	A default label shall appear as either the first or the last switch label of a switch statement.	required	-	yes
16.6	Every switch statement shall have at least two switch-clauses.	required	-	yes

16.7	A switch-expression shall not have essentially Boolean type.	required	-	yes
17.1	The features of <stdarg.h> shall not be used.	required	-	yes
17.2	Functions shall not call themselves, either directly or indirectly.	required	-	yes
17.3	A function shall not be declared implicitly.	mandatory	-	yes
17.4	All exit paths from a function with non-void return type shall have an explicit return statement with an expression.	mandatory	-	yes
17.5	The function argument corresponding to a parameter declared to have an array type shall have an appropriate number of elements.	advisory	-	no
17.6	The declaration of an array parameter shall not contain the static keyword between the [].	mandatory	-	yes
17.7	The value returned by a function having non-void return type shall be used.	required	-	yes
17.8	A function parameter should not be modified.	advisory	-	no
18.1	A pointer resulting from arithmetic on a pointer operand shall address an element of the same array as that pointer operand.	required	-	yes
18.2	Subtraction between pointers shall only be applied to pointers that address elements of the same array.	required	-	yes
18.3	The relational operators >, >=, < and <= shall not be applied to objects of pointer type except where they point into the same object.	required	-	yes
18.4	The +, -, += and -= operators should not be applied to an expression of pointer type.	advisory	-	no
18.5	Declarations should contain no more than two levels of pointer nesting.	advisory	-	no
18.6	The address of an object with automatic storage shall not be copied to another object that persists after the first object has ceased to exist.	required	-	yes
18.7	Flexible array members shall not be declared.	required	-	yes
18.8	Variable-length array types shall not be used.	required	-	yes
19.1	An object shall not be assigned or copied to an overlapping object.	mandatory	-	yes
19.2	The union keyword should not be used.	advisory	-	no
20.1	#include directives should only be preceded by preprocessor directives or comments.	advisory	-	no
20.2	The ', " or \ characters and the /* or // character sequences shall not occur in a header file name.	required	-	yes
20.3	The #include directive shall be followed by either a <filename> or "filename"sequence.	required	-	yes
20.4	A macro shall not be defined with the same name as a keyword.	required	-	yes

20.5	#undef should not be used.	advisory	-	no
20.6	Tokens that look like a preprocessing directive shall not occur within a macro argument.	required	-	yes
20.7	Expressions resulting from the expansion of macro parameters shall be enclosed in parentheses.	required	-	yes
20.8	The controlling expression of a #if or #elif preprocessing directive shall evaluate to 0 or 1.	required	-	yes
20.9	All identifiers used in the controlling expression of #if or #elif preprocessing directives shall be #define'd before evaluation.	required	-	yes
20.10	The # and ## preprocessor operators should not be used.	advisory	-	no
20.11	A macro parameter immediately following a # operator shall not immediately be followed by a ## operator.	required	-	yes
20.12	A macro parameter used as an operand to the # or ## operators, which is itself subject to further macro replacement, shall only be used as an operand to these operators.	required	-	yes
20.13	A line whose first token is # shall be a valid preprocessing directive.	required	-	yes
20.14	All #else, #elif and #endif preprocessor directives shall reside in the same file as the #if, #ifdef or #ifndef directive to which they are related.	required	-	yes
21.1	#define and #undef shall not be used on a reserved identifier or reserved macro name.	required	-	yes
21.2	A reserved identifier or macro name shall not be declared.	required	-	yes
21.3	The memory allocation and deallocation functions of <stdlib.h> shall not be used.	required	-	yes
21.4	The standard header file <setjmp.h> shall not be used.	required	-	yes
21.5	The standard header file <signal.h> shall not be used.	required	-	yes
21.6	The Standard Library input/output functions shall not be used.	required	-	yes
21.7	The atof, atoi, atol, and atoll functions of <stdlib.h> shall not be used.	required	-	yes
21.8	The library functions abort, exit and system of <stdlib.h> shall not be used.	required	-	yes
21.9	The library functions bsearch and qsort of <stdlib.h> shall not be used.	required	-	yes
21.10	The Standard Library time and date functions shall not be used.	required	-	yes
21.11	The standard header file <tgmath.h> shall not be used.	required	-	yes
21.12	The exception handling features of <fenv.h> should not be used.	advisory	-	no
21.13	Any value passed to a function in <ctype.h> shall be representable as an unsigned char or be the value EOF.	mandatory	-	yes
21.14	The Standard Library function memcmp shall not be used to compare null terminated strings.	required	-	yes

21.15	The pointer arguments to the Standard Library functions memcpy, memmove and memcmp shall be pointers to qualified or unqualified versions of compatible types.	required	-	yes
21.16	The pointer arguments to the Standard Library function memcmp shall point to either a pointer type, an essentially signed type, an essentially unsigned type, an essentially Boolean type or an essentially enum type.	required	-	yes
21.17	Use of the string handling functions from <string.h> shall not result in accesses beyond the bounds of the objects referenced by their pointer parameters.	mandatory	-	yes
21.18	The size_t argument passed to any function in <string.h> shall have an appropriate value.	mandatory	-	yes
21.19	The pointers returned by the Standard Library functions localeconv, getenv, setlocale or, strerror shall only be used as if they have pointer to const-qualified type.	mandatory	-	yes
21.20	The pointer returned by the Standard Library functions asctime, ctime, gmtime, localtime, localeconv, getenv, setlocale or strerror shall not be used following a subsequent call to the same function.	mandatory	-	yes
22.1	All resources obtained dynamically by means of Standard Library functions shall be explicitly released.	required	-	yes
22.2	A block of memory shall only be freed if it was allocated by means of a Standard Library function.	mandatory	-	yes
22.3	The same file shall not be open for read and write access at the same time on different streams.	required	-	yes
22.4	There shall be no attempt to write to a stream which has been opened as read-only.	mandatory	-	yes
22.5	A pointer to a FILE object shall not be dereferenced.	mandatory	-	yes
22.6	The value of a pointer to a FILE shall not be used after the associated stream has been closed.	mandatory	-	yes
22.7	The macro EOF shall only be compared with the unmodified return value from any Standard Library function capable of returning EOF.	required	-	yes
22.8	The value of errno shall be set to zero prior to a call to an errno-setting-function.	required	-	yes
22.9	The value of errno shall be tested against zero after calling an errno-setting-function.	required	-	yes
22.10	The value of errno shall only be tested when the last function to be called was an errno-setting-function.	required	-	yes

Chapter 5. Appendix 2 - Definitions

Table 5.1. Abbreviations

Abbreviation	Definition
NA	Not Available