MISRA C++ 2008 Mapping to CodeSonar®

Category ID	Category Name	CodeSonar Class Mnemonic	CodeSonar Class Name	Relationship Type (category to class)
MisraC++2008:0-1-1	A project shall not contain unreachable code.	LANG.STRUCT.UC	Unreachable Call	closely mapped
MisraC++2008:0-1-1	A project shall not contain unreachable code.	LANG.STRUCT.UC	Unreachable Computation	closely mapped
MisraC++2008:0-1-1	A project shall not contain unreachable code.	LANG.STRUCT.UC	Unreachable Conditional	closely mapped
MisraC++2008:0-1-1	A project shall not contain unreachable code.	LANG.STRUCT.UC	Unreachable Control Flow	closely mapped
MisraC++2008:0-1-1	A project shall not contain unreachable code.	LANG.STRUCT.UC	Unreachable Data Flow	closely mapped
MisraC++2008:0-1-3	A project shall not contain unused variables.	LANG.STRUCT.UUVAR	Unused Variable	closely mapped
MisraC++2008:0-1-5	A project shall not contain unused type declarations.	LANG.STRUCT.UUTYPE	Unused Type	closely mapped
MisraC++2008:0-1-6	A project shall not contain instances of non-volatile variables being given values that are never subsequently used.	LANG.STRUCT.UUVAL	Unused Value	closely mapped
MisraC++2008:0-1-7	The value returned by a function having a non-void return type that is not an overloaded operator shall always be used.	LANG.FUNCS.IRV	Ignored Return Value	closely mapped
MisraC++2008:0-1-8	All functions with void return type shall have external side effect(s).	MISC.NOEFFECT	Function Call Has No Effect	closely mapped
MisraC++2008:0-1-9	There shall be no dead code.	MISC.NOEFFECT	Function Call Has No Effect	closely mapped
MisraC++2008:0-1-9	There shall be no dead code.	LANG.STRUCT.UUVAL	Unused Value	closely mapped
MisraC++2008:0-1-9	There shall be no dead code.	LANG.STRUCT.UA	Useless Assignment	closely mapped
MisraC++2008:0-1-11	There shall be no unused parameters (named or unnamed) in non-virtual functions.	LANG.STRUCT.UUPARAM	Unused Parameter	closely mapped
MisraC++2008:0-1-12	There shall be no unused parameters (named or unnamed) in the set of parameters for a virtual function and all the functions that override it.	LANG.STRUCT.UUPARAM	Unused Parameter	closely mapped
MisraC++2008:0-2-1	An object shall not be assigned to an overlapping object.	MISC.MEM.OR	Overlapping Memory Regions	closely mapped
MisraC++2008:0-3-2	If a function generates error information, then that error information shall be tested.	LANG.FUNCS.IRV	Ignored Return Value	closely mapped
MisraC++2008:2-3-1	Trigraphs shall not be used.	LANG.STRUCT.TRIGRAPH	Trigraph	closely mapped

MISRA C++ 2008 Mapping to CodeSonar®

MisraC++2008:2-7-1	The character sequence /* shall not be used within a C-style comment.	LANG.COMM.NEST.CSTYLE	/* in Comment	closely mapped
MisraC++2008:2-7-2	Sections of code shall not be "commented out" using C-style comments.	LANG.COMM.CODE	Commented-out Code	closely mapped
MisraC++2008:2-7-3	Sections of code should not be "commented out" using C++ comments.	LANG.COMM.CODE	Commented-out Code	closely mapped
MisraC++2008:2-10-1	Different identifiers shall be typographically unambiguous.	LANG.ID.AMBIG	Typographically Ambiguous Identifiers	closely mapped
MisraC++2008:2-10-2	Identifiers declared in an inner scope shall not hide an identifier declared in an outer scope.	LANG.ID.ND.NEST	Non-distinct Identifiers: Nested Scope	closely mapped
MisraC++2008:2-10-3	A typedef name (including qualification, if any) shall be a unique identifier.	LANG.ID.NU.TYPE	Non-unique Identifiers: Typedef	closely mapped
MisraC++2008:2-10-4	A class, union or enum name (including qualification, if any) shall be a unique identifier.	LANG.ID.NU.TAG	Non-unique Identifiers: Tag	closely mapped
MisraC++2008:2-10-5	The identifier name of a non-member object or function with static storage duration should not be reused.	LANG.ID.NU.INT	Non-unique Identifiers: Internal Name	closely mapped
MisraC++2008:2-10-6	If an identifier refers to a type, it shall not also refer to an object or a function in the same scope.	LANG.ID.ND.SS	Non-distinct Identifiers: Same Scope	closely mapped
MisraC++2008:2-13-2	Octal constants (other than zero) and octal escape sequences (other than "\0") shall not be used.	LANG.TYPE.OC	Octal Constant	closely mapped
MisraC++2008:2-13-3	A "U" suffix shall be applied to all octal or hexadecimal integer literals of unsigned type.	LANG.TYPE.MSUF	Missing Literal Suffix	closely mapped
MisraC++2008:2-13-4	Literal suffixes shall be upper case.	LANG.TYPE.CSUF	Confusing Literal Suffix	closely mapped
MisraC++2008:3-1-1	It shall be possible to include any header file in multiple translation units without violating the One Definition Rule.	LANG.STRUCT.DEF.FDH	Function Defined in Header File	closely mapped
MisraC++2008:3-1-1	It shall be possible to include any header file in multiple translation units without violating the One Definition Rule.	LANG.STRUCT.DEF.ODH	Object Defined in Header File	closely mapped
MisraC++2008:3-1-2	Functions shall not be declared at block scope.	LANG.STRUCT.DECL.FNEST	Nested Function Declaration	closely mapped

MISRA C++ 2008 Mapping to CodeSonar®

MisraC++2008:3-1-3	When an array is declared, its size shall either be stated explicitly or defined implicitly by initialization.	LANG.STRUCT.DECL.EAWS	Extern Array Without Size	closely mapped
MisraC++2008:3-2-1	All declarations of an object or function shall have compatible types.	LANG.STRUCT.DECL.MGT	Global Variable Declared with Different Types	closely mapped
MisraC++2008:3-2-1	All declarations of an object or function shall have compatible types.	LANG.STRUCT.DECL.IF	Inconsistent Function Declarations	closely mapped
MisraC++2008:3-2-1	All declarations of an object or function shall have compatible types.	LANG.STRUCT.DECL.IO	Inconsistent Object Declarations	closely mapped
MisraC++2008:3-2-2	The One Definition Rule shall not be violated.	LANG.STRUCT.DEF.NOEXT	Missing External Definition	closely mapped
MisraC++2008:3-2-2	The One Definition Rule shall not be violated.	LANG.STRUCT.DEF.MULTIEXT	Multiple External Definitions	closely mapped
MisraC++2008:3-2-3	A type, object or function that is used in multiple translation units shall be declared in one and only one file.	LANG.STRUCT.DECL.NOEXT	Missing External Declaration	closely mapped
MisraC++2008:3-2-3	A type, object or function that is used in multiple translation units shall be declared in one and only one file.	LANG.STRUCT.DECL.MG	Multiple Declarations of a Global	closely mapped
MisraC++2008:3-2-3	A type, object or function that is used in multiple translation units shall be declared in one and only one file.	LANG.STRUCT.DECL.MULTIEXT	Multiple External Declarations	closely mapped
MisraC++2008:3-2-4	An identifier with external linkage shall have exactly one definition.	LANG.STRUCT.DEF.NOEXT	Missing External Definition	closely mapped
MisraC++2008:3-2-4	An identifier with external linkage shall have exactly one definition.	LANG.STRUCT.DEF.MULTIEXT	Multiple External Definitions	closely mapped
MisraC++2008:3-3-2	If a function has internal linkage then all re-declarations shall include the static storage class specifier.	LANG.STRUCT.SCOPE.FILE	Scope Could Be File Static	closely mapped
MisraC++2008:3-3-2	If a function has internal linkage then all re-declarations shall include the static storage class specifier.	LANG.STRUCT.SCOPE.LOCAL	Scope Could Be Local Static	closely mapped
MisraC++2008:3-4-1	An identifier declared to be an object or type shall be defined in a block that minimizes its visibility.	LANG.STRUCT.SCOPE.LOCAL	Scope Could Be Local Static	closely mapped

MisraC++2008:3-9-1	The types used for an object, a function return type, or a function parameter shall be token-for-token identical in all declarations and re-declarations.	LANG.STRUCT.DECL.MGT	Global Variable Declared with Different Types	closely mapped
MisraC++2008:3-9-1	The types used for an object, a function return type, or a function parameter shall be token-for-token identical in all declarations and re-declarations.	LANG.STRUCT.DECL.IF	Inconsistent Function Declarations	closely mapped
MisraC++2008:3-9-1	The types used for an object, a function return type, or a function parameter shall be token-for-token identical in all declarations and re-declarations.	LANG.STRUCT.DECL.IO	Inconsistent Object Declarations	closely mapped
MisraC++2008:3-9-2	typedefs that indicate size and signedness should be used in place of the basic numerical types.	LANG.TYPE.BASIC	Basic Numerical Type Used	closely mapped
MisraC++2008:4-5-1	Expressions with type bool shall not be used as operands to built-in operators other than the assignment operator = , the logical operators && , || , ! , the equality operators == and != , the unary & operator, and the conditional operator.	LANG.TYPE.IOT	Inappropriate Operand Type	closely mapped
MisraC++2008:4-5-2	Expressions with type enum shall not be used as operands to built-in operators other than the subscript operator [ ] , the assignment operator = , the equality operators == and != , the unary & operator, and the relational operators < , <= , > , >= .	LANG.TYPE.IOT	Inappropriate Operand Type	closely mapped
MisraC++2008:4-5-3	Expressions with type (plain) char and wchar_t shall not be used as operands to built-in operators other than the assignment operator = , the equality operators == and != , and the unary & operator.	LANG.TYPE.ICA	Inappropriate Character Arithmetic	closely mapped

MisraC++2008:4-5-3	Expressions with type (plain) char and wchar_t shall not be used as operands to built-in operators other than the assignment operator = , the equality operators == and != , and the unary & operator.	LANG.TYPE.IOT	Inappropriate Operand Type	closely mapped
MisraC++2008:5-0-3	A cvalue expression shall not be implicitly converted to a different underlying type.	LANG.TYPE.AWID	Expression Value Widened by Assignment	closely mapped
MisraC++2008:5-0-3	A cvalue expression shall not be implicitly converted to a different underlying type.	LANG.TYPE.OWID	Expression Value Widened by Other Operand	closely mapped
MisraC++2008:5-0-3	A cvalue expression shall not be implicitly converted to a different underlying type.	LANG.TYPE.IAT	Inappropriate Assignment Type	closely mapped
MisraC++2008:5-0-3	A cvalue expression shall not be implicitly converted to a different underlying type.	LANG.TYPE.MOT	Mismatched Operand Types	closely mapped
MisraC++2008:5-0-4	An implicit integral conversion shall not change the signedness of the underlying type.	LANG.CAST.COERCE	Coercion Alters Value	closely mapped
MisraC++2008:5-0-5	There shall be no implicit floating-integral conversions.	LANG.TYPE.AWID	Expression Value Widened by Assignment	closely mapped
MisraC++2008:5-0-5	There shall be no implicit floating-integral conversions.	LANG.TYPE.OWID	Expression Value Widened by Other Operand	closely mapped
MisraC++2008:5-0-5	There shall be no implicit floating-integral conversions.	LANG.TYPE.IAT	Inappropriate Assignment Type	closely mapped
MisraC++2008:5-0-5	There shall be no implicit floating-integral conversions.	LANG.TYPE.MOT	Mismatched Operand Types	closely mapped
MisraC++2008:5-0-6	An implicit integral or floating-point conversion shall not reduce the size of the underlying type.	LANG.CAST.COERCE	Coercion Alters Value	closely mapped
MisraC++2008:5-0-6	An implicit integral or floating-point conversion shall not reduce the size of the underlying type.	LANG.TYPE.IAT	Inappropriate Assignment Type	closely mapped
MisraC++2008:5-0-7	There shall be no explicit floating-integral conversions of a cvalue expression.	LANG.TYPE.ICTE	Inappropriate Cast Type: Expression	closely mapped
MisraC++2008:5-0-8	An explicit integral or floating-point conversion shall not increase the size of the underlying type of a cvalue expression.	LANG.TYPE.ICTE	Inappropriate Cast Type: Expression	closely mapped

MisraC++2008:5-0-9	An explicit integral conversion shall not change the signedness of the underlying type of a cvalue expression.	LANG.CAST.VALUE	Cast Alters Value	closely mapped
MisraC++2008:5-0-10	If the bitwise operators ~ and << are applied to an operand with an underlying type of unsigned char or unsigned short, the result shall be immediately cast to the underlying type of the operand.	LANG.CAST.RIP	Risky Integer Promotion	closely mapped
MisraC++2008:5-0-11	The plain char type shall only be used for the storage and use of character values.	LANG.TYPE.IAT	Inappropriate Assignment Type	closely mapped
MisraC++2008:5-0-11	The plain char type shall only be used for the storage and use of character values.	LANG.TYPE.ICA	Inappropriate Character Arithmetic	closely mapped
MisraC++2008:5-0-11	The plain char type shall only be used for the storage and use of character values.	LANG.TYPE.IOT	Inappropriate Operand Type	closely mapped
MisraC++2008:5-0-11	The plain char type shall only be used for the storage and use of character values.	LANG.TYPE.MOT	Mismatched Operand Types	closely mapped
MisraC++2008:5-0-12	signed char and unsigned char type shall only be used for the storage and use of numeric values.	LANG.TYPE.IAT	Inappropriate Assignment Type	closely mapped
MisraC++2008:5-0-12	signed char and unsigned char type shall only be used for the storage and use of numeric values.	LANG.TYPE.ICA	Inappropriate Character Arithmetic	closely mapped
MisraC++2008:5-0-12	signed char and unsigned char type shall only be used for the storage and use of numeric values.	LANG.TYPE.IOT	Inappropriate Operand Type	closely mapped
MisraC++2008:5-0-12	signed char and unsigned char type shall only be used for the storage and use of numeric values.	LANG.TYPE.MOT	Mismatched Operand Types	closely mapped
MisraC++2008:5-0-13	The condition of an if-statement and the condition of an iteration-statement shall have type bool.	LANG.STRUCT.NBC	Condition Is Not Boolean	closely mapped
MisraC++2008:5-0-14	The first operand of a conditional-operator shall have type bool.	LANG.TYPE.IOT	Inappropriate Operand Type	closely mapped

MisraC++2008:5-0-16	A pointer operand and any pointer resulting from pointer arithmetic using that operand shall both address elements of the same array.	LANG.MEM.BO	Buffer Overrun	also related
MisraC++2008:5-0-16	A pointer operand and any pointer resulting from pointer arithmetic using that operand shall both address elements of the same array.	LANG.MEM.BU	Buffer Underrun	also related
MisraC++2008:5-0-16	A pointer operand and any pointer resulting from pointer arithmetic using that operand shall both address elements of the same array.	LANG.STRUCT.PBB	Pointer Before Beginning of Object	also related
MisraC++2008:5-0-16	A pointer operand and any pointer resulting from pointer arithmetic using that operand shall both address elements of the same array.	LANG.STRUCT.PPE	Pointer Past End of Object	also related
MisraC++2008:5-0-16	A pointer operand and any pointer resulting from pointer arithmetic using that operand shall both address elements of the same array.	LANG.MEM.TBA	Tainted Buffer Access	also related
MisraC++2008:5-0-16	A pointer operand and any pointer resulting from pointer arithmetic using that operand shall both address elements of the same array.	LANG.MEM.TO	Type Overrun	also related
MisraC++2008:5-0-16	A pointer operand and any pointer resulting from pointer arithmetic using that operand shall both address elements of the same array.	LANG.MEM.TU	Type Underrun	also related
MisraC++2008:5-0-20	Non-constant operands to a binary bitwise operator shall have the same underlying type.	LANG.TYPE.IOT	Inappropriate Operand Type	closely mapped
MisraC++2008:5-0-21	Bitwise operators shall only be applied to operands of unsigned underlying type.	LANG.TYPE.IOT	Inappropriate Operand Type	closely mapped
MisraC++2008:5-2-4	MisraC++2008:5-2-4	LANG.CAST.VALUE	Cast Alters Value	also related
MisraC++2008:5-2-4	MisraC++2008:5-2-4	LANG.CAST.FN	Dangerous Function Cast	also related
MisraC++2008:5-2-5	A cast shall not remove any const or volatile qualification from the type of a pointer or reference.	LANG.CAST.PC.CRCQ	Cast Removes const Qualifier	closely mapped

MisraC++2008:5-2-5	A cast shall not remove any const or volatile qualification from the type of a pointer or reference.	LANG.CAST.PC.CRVQ	Cast Removes volatile Qualifier	closely mapped
MisraC++2008:5-2-6	A cast shall not convert a pointer to a function to any other pointer type, including a pointer to function type.	LANG.CAST.FN	Dangerous Function Cast	closely mapped
MisraC++2008:5-2-6	A cast shall not convert a pointer to a function to any other pointer type, including a pointer to function type.	LANG.STRUCT.FUNCPTR.CONVERT	Function Pointer Conversion	also related
MisraC++2008:5-2-7	An object with pointer type shall not be converted to an unrelated pointer type, either directly or indirectly.	LANG.CAST.PC.OBJ	Cast: Object Pointers	closely mapped
MisraC++2008:5-2-7	An object with pointer type shall not be converted to an unrelated pointer type, either directly or indirectly.	LANG.CAST.PC.FN2DATA	Conversion from Function Pointer	closely mapped
MisraC++2008:5-2-7	An object with pointer type shall not be converted to an unrelated pointer type, either directly or indirectly.	LANG.CAST.PC.INC	Conversion: Pointer to Incomplete	closely mapped
MisraC++2008:5-2-7	An object with pointer type shall not be converted to an unrelated pointer type, either directly or indirectly.	LANG.CAST.PC.INT	Conversion: Pointer/Integer	closely mapped
MisraC++2008:5-2-7	An object with pointer type shall not be converted to an unrelated pointer type, either directly or indirectly.	LANG.CAST.PC.PV	Conversion: Void Pointer to Object Pointer	closely mapped
MisraC++2008:5-2-8	An object with integer type or pointer to void type shall not be converted to an object with pointer type.	LANG.CAST.PC.INT	Conversion: Pointer/Integer	closely mapped
MisraC++2008:5-2-8	An object with integer type or pointer to void type shall not be converted to an object with pointer type.	LANG.CAST.PC.PV	Conversion: Void Pointer to Object Pointer	closely mapped
MisraC++2008:5-2-9	A cast should not convert a pointer type to an integral type.	LANG.CAST.PC.INT	Conversion: Pointer/Integer	closely mapped
MisraC++2008:5-2-10	The increment (++) and decrement (--) operators should not be mixed with other operators in an expression.	LANG.STRUCT.SE.DEC	Side Effects in Expression with Decrement	closely mapped
MisraC++2008:5-2-10	The increment (++) and decrement (--) operators should not be mixed with other operators in an expression.	LANG.STRUCT.SE.INC	Side Effects in Expression with Increment	closely mapped

MISRA C++ 2008 Mapping to CodeSonar®

MisraC++2008:5-3-1	Each operand of the ! operator, the logical && or the logical || operators shall have type bool.	LANG.TYPE.IOT	Inappropriate Operand Type	closely mapped
MisraC++2008:5-3-2	The unary minus operator shall not be applied to an expression whose underlying type is unsigned.	LANG.TYPE.IOT	Inappropriate Operand Type	closely mapped
MisraC++2008:5-3-4	Evaluation of the operand to the sizeof operator shall not contain side effects.	LANG.STRUCT.SE.SIZEOF	Side Effects in sizeof	closely mapped
MisraC++2008:5-8-1	The right hand operand of a shift operator shall lie between zero and one less than the width in bits of the underlying type of the left hand operand.	LANG.ARITH.NEGSHIFT	Negative Shift Amount	closely mapped
MisraC++2008:5-8-1	The right hand operand of a shift operator shall lie between zero and one less than the width in bits of the underlying type of the left hand operand.	LANG.ARITH.BIGSHIFT	Shift Amount Exceeds Bit Width	closely mapped
MisraC++2008:6-3-1	The statement forming the body of a switch, while, do ... while or for statement shall be a compound statement.	LANG.STRUCT.BNC	Body Is Not Compound Statement	closely mapped
MisraC++2008:6-4-1	An if (condition) construct shall be followed by a compound statement. The else keyword shall be followed by either a compound statement, or another if statement.	LANG.STRUCT.BNC	Body Is Not Compound Statement	closely mapped
MisraC++2008:6-4-2	All if ... else if constructs shall be terminated with an else clause.	LANG.STRUCT.NOELSE	Missing Final else	closely mapped
MisraC++2008:6-4-3	A switch statement shall be a well-formed switch statement.	LANG.STRUCT.SW.BAD	Malformed switch Statement	closely mapped
MisraC++2008:6-4-5	An unconditional throw or break statement shall terminate every non-empty switch-clause.	LANG.STRUCT.SW.MB	Missing break	closely mapped
MisraC++2008:6-4-6	The final clause of a switch statement shall be the default-clause.	LANG.STRUCT.SW.MPD	Misplaced default	closely mapped
MisraC++2008:6-4-6	The final clause of a switch statement shall be the default-clause.	LANG.STRUCT.SW.MD	Missing default	closely mapped
MisraC++2008:6-4-7	The condition of a switch statement shall not have bool type.	LANG.STRUCT.SW.BOOL	Boolean switch Expression	closely mapped
MisraC++2008:6-4-8	Every switch statement shall have at least one case-clause.	LANG.STRUCT.SW.IF	Too Few Cases in switch	closely mapped

MISRA C++ 2008 Mapping to CodeSonar®

MisraC++2008:6-5-1	A for loop shall contain a single loop-counter which shall not have floating type.	LANG.STRUCT.LOOP.FPC	Float-typed Loop Counter	closely mapped
MisraC++2008:6-5-3	The loop-counter shall not be modified within condition or statement.	LANG.STRUCT.LOOP.MFTERM	Malformed for-loop Condition	closely mapped
MisraC++2008:6-5-3	The loop-counter shall not be modified within condition or statement.	LANG.STRUCT.LOOP.MFINIT	Malformed for-loop Initialization	closely mapped
MisraC++2008:6-5-4	The loop-counter shall be modified by one of -- , ++ , -=n , or +=n; where n remains constant for the duration of the loop.	LANG.STRUCT.LOOP.MFSTEP	Malformed for-loop Step	closely mapped
MisraC++2008:6-5-5	A loop-control-variable other than the loop-counter shall not be modified within condition or expression.	LANG.STRUCT.LOOP.MFTERM	Malformed for-loop Condition	closely mapped
MisraC++2008:6-5-5	A loop-control-variable other than the loop-counter shall not be modified within condition or expression.	LANG.STRUCT.LOOP.MFSTEP	Malformed for-loop Step	closely mapped
MisraC++2008:6-6-1	Any label referenced by a goto statement shall be declared in the same block, or in a block enclosing the goto statement.	LANG.STRUCT.GLABEL	Label Not In Enclosing Block	closely mapped
MisraC++2008:6-6-2	The goto statement shall jump to a label declared later in the same function body.	LANG.STRUCT.BGOTO	Backwards goto	closely mapped
MisraC++2008:6-6-3	The continue statement shall only be used within a well-formed for loop.	LANG.STRUCT.CONTINUE	Continue Statement	closely mapped
MisraC++2008:6-6-4	For any iteration statement there shall be no more than one break or goto statement used for loop termination.	LANG.STRUCT.LOOP.MAE	Multiple Abnormal Loop Exits	closely mapped
MisraC++2008:6-6-5	A function shall have a single point of exit at the end of the function.	LANG.STRUCT.MISRS	Misplaced Return Statement	closely mapped
MisraC++2008:6-6-5	A function shall have a single point of exit at the end of the function.	LANG.STRUCT.MULRS	Multiple Return Statements	closely mapped
MisraC++2008:7-4-3	Assembly language shall be encapsulated and isolated.	LANG.ASM.MIXED	Mixed Assembly and Code	closely mapped
MisraC++2008:7-5-1	A function shall not return a reference or a pointer to an automatic variable (including parameters), defined within the function.	LANG.STRUCT.RPL	Return Pointer to Local	closely mapped

MisraC++2008:7-5-4	Functions should not call themselves, either directly or indirectly.	LANG.FUNCS.RECURSION	Recursion	closely mapped
MisraC++2008:8-4-2	The identifiers used for the parameters in a re-declaration of a function shall be identical to those in the declaration.	LANG.STRUCT.DECL.MGT	Global Variable Declared with Different Types	hierarchy descendant
MisraC++2008:8-4-2	The identifiers used for the parameters in a re-declaration of a function shall be identical to those in the declaration.	LANG.STRUCT.DECL.IF	Inconsistent Function Declarations	closely mapped
MisraC++2008:8-4-2	The identifiers used for the parameters in a re-declaration of a function shall be identical to those in the declaration.	LANG.STRUCT.DECL.IO	Inconsistent Object Declarations	hierarchy descendant
MisraC++2008:8-4-3	All exit paths from a function with non-void return type shall have an explicit return statement with an expression.	LANG.STRUCT.MRS	Missing Return Statement	closely mapped
MisraC++2008:8-5-1	All variables shall have a defined value before they are used.	LANG.MEM.UVAR	Uninitialized Variable	closely mapped
MisraC++2008:8-5-2	Braces shall be used to indicate and match the structure in the non-zero initialization of arrays and structures.	LANG.STRUCT.INIT.MBI	Missing Braces in Initialization	closely mapped
MisraC++2008:8-5-3	In an enumerator list, the = construct shall not be used to explicitly initialize members other than the first, unless all items are explicitly initialized.	LANG.STRUCT.INIT.ENUM	Inconsistent Enumerator Initialization	closely mapped
MisraC++2008:9-5-1	Unions shall not be used.	LANG.TYPE.UNION	Union Type	closely mapped
MisraC++2008:9-6-2	Bit-fields shall be either bool type or an explicitly unsigned or signed integral type.	LANG.TYPE.BFSIGN	Bit-field Signedness Not Explicit	closely mapped
MisraC++2008:9-6-2	Bit-fields shall be either bool type or an explicitly unsigned or signed integral type.	LANG.TYPE.BFINT	Inappropriate Bit-field Type	closely mapped
MisraC++2008:9-6-3	Bit-fields shall not have enum type.	LANG.TYPE.BFSIGN	Bit-field Signedness Not Explicit	closely mapped
MisraC++2008:9-6-3	Bit-fields shall not have enum type.	LANG.TYPE.BFINT	Inappropriate Bit-field Type	closely mapped
MisraC++2008:9-6-4	Named bit-fields with signed integer type shall have a length of more than one bit.	LANG.TYPE.BFSHORT	Bit-field Too Short	closely mapped

MISRA C++ 2008 Mapping to CodeSonar®

MisraC++2008:16-0-1	#include directives in a file shall only be preceded by preprocessor directives or comments.	LANG.PREPROC.CBI	Code Before #include	closely mapped
MisraC++2008:16-0-3	#undef shall not be used.	LANG.PREPROC.RUNDEF	Macro Undefinition of Reserved Name	closely mapped
MisraC++2008:16-0-3	#undef shall not be used.	LANG.PREPROC.UNDEF	Use of #undef	closely mapped
MisraC++2008:16-0-5	Arguments to a function-like macro shall not contain tokens that look like preprocessing directives.	LANG.PREPROC.MACROARG	Preprocessing Directives in Macro Argument	closely mapped
MisraC++2008:16-0-8	If the # token appears as the first token on a line, then it shall be immediately followed by a preprocessing token.	LANG.PREPROC.INVALID	Invalid Preprocessor Directive	closely mapped
MisraC++2008:16-1-2	All #else, #elif and #endif preprocessor directives shall reside in the same file as the #if or #ifdef directive to which they are related.	LANG.PREPROC.NOENDIF	No Matching #endif	closely mapped
MisraC++2008:16-1-2	All #else, #elif and #endif preprocessor directives shall reside in the same file as the #if or #ifdef directive to which they are related.	LANG.PREPROC.NOIF	No Matching #if	closely mapped
MisraC++2008:16-2-4	The ', ", /* or // characters shall not occur in a header file name.	LANG.PREPROC.INCL.FNAME	Dangerous Include File Name	closely mapped
MisraC++2008:16-2-5	The \ character should not occur in a header file name.	LANG.PREPROC.INCL.FNAME	Dangerous Include File Name	closely mapped
MisraC++2008:16-2-6	The #include directive shall be followed by either a <filename> or "filename" sequence.	LANG.PREPROC.INCL.MF	Malformed #include	closely mapped
MisraC++2008:16-3-2	The # and ## operators should not be used.	LANG.PREPROC.PASTE	Macro Uses ## Operator	closely mapped
MisraC++2008:17-0-1	Reserved identifiers, macros and functions in the standard library shall not be defined, redefined or undefined.	LANG.PREPROC.RDEF	Macro Definition of Reserved Name	closely mapped
MisraC++2008:17-0-1	Reserved identifiers, macros and functions in the standard library shall not be defined, redefined or undefined.	LANG.PREPROC.RUNDEF	Macro Undefinition of Reserved Name	closely mapped
MisraC++2008:17-0-5	The setjmp macro and the longjmp function shall not be used.	BADFUNC.LONGJMP	Use of longjmp	closely mapped
MisraC++2008:17-0-5	The setjmp macro and the longjmp function shall not be used.	BADFUNC.SETJMP	Use of setjmp	closely mapped

MisraC++2008:18-0-1	The C library shall not be used.	LANG.PREPROC.INCL.SETJMP_H	Use of <setjmp.h>	closely mapped
MisraC++2008:18-0-1	The C library shall not be used.	LANG.PREPROC.INCL.SIGNAL_H	Use of <signal.h>	closely mapped
MisraC++2008:18-0-1	The C library shall not be used.	LANG.PREPROC.INCL.TGMATH_H	Use of <tgmath.h>	closely mapped
MisraC++2008:18-0-2	The library functions atof, atoi and atol from library <cstdlib> shall not be used.	BADFUNC.ATOF	Use of atof	closely mapped
MisraC++2008:18-0-2	The library functions atof, atoi and atol from library <cstdlib> shall not be used.	BADFUNC.ATOI	Use of atoi	closely mapped
MisraC++2008:18-0-2	The library functions atof, atoi and atol from library <cstdlib> shall not be used.	BADFUNC.ATOL	Use of atol	closely mapped
MisraC++2008:18-0-2	The library functions atof, atoi and atol from library <cstdlib> shall not be used.	BADFUNC.ATOLL	Use of atoll	closely mapped
MisraC++2008:18-0-3	The library functions abort, exit, getenv and system from library <cstdlib> shall not be used.	BADFUNC.ABORT	Use of abort	closely mapped
MisraC++2008:18-0-3	The library functions abort, exit, getenv and system from library <cstdlib> shall not be used.	BADFUNC.EXIT	Use of exit	closely mapped
MisraC++2008:18-0-3	The library functions abort, exit, getenv and system from library <cstdlib> shall not be used.	BADFUNC.GETENV	Use of getenv	closely mapped
MisraC++2008:18-0-3	The library functions abort, exit, getenv and system from library <cstdlib> shall not be used.	BADFUNC.PATH.SYSTEM	Use of system	closely mapped
MisraC++2008:18-0-4	The time handling functions of library <ctime> shall not be used.	BADFUNC.TIME_H	Use of <time.h> Time/Date Function	closely mapped
MisraC++2008:18-0-5	The unbounded functions of library <cstring> shall not be used.	BADFUNC.BO.STRCMP	Use of strcmp	closely mapped
MisraC++2008:18-0-5	The unbounded functions of library <cstring> shall not be used.	BADFUNC.BO.STRCPY	Use of strcpy	closely mapped
MisraC++2008:18-0-5	The unbounded functions of library <cstring> shall not be used.	BADFUNC.BO.STRLEN	Use of strlen	closely mapped
MisraC++2008:18-2-1	The macro offsetof shall not be used.	BADMACRO.OFFSETOF	Use of offsetof	closely mapped
MisraC++2008:18-4-1	Dynamic heap memory allocation shall not be used.	ALLOC.POSTINIT	Dynamic Allocation After Initialization	closely mapped
MisraC++2008:18-7-1	The signal handling facilities of <csignal> shall not be used.	BADFUNC.SIGNAL	Use of signal	closely mapped

MisraC++2008:27-0-1	The stream input/output library <cstdio> shall not be used.	BADFUNC.STDIO_H	Use of <stdio.h> Input/Output	closely mapped
MisraC++2008:27-0-1	The stream input/output library <cstdio> shall not be used.	BADFUNC.WCHAR_H	Use of <wchar.h> Input/Output	closely mapped