|  |  |
| --- | --- |
| D:\AMBIENTE\Desktop\CGS_OHB.gif | **NEOSTEL** |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Tipo Doc.:  *Doc.Type:* | Technical Note | | | | | N° DRD:  *DRD N°:* |  | | | |
| N° Doc.:  *Doc. N°:* | **NEOSTEL-TN-CGS-001** | Ediz.:  *Issue:* |  | Data:  *Date:* |  | | Pagina  *Page* | **1** | Di  *Of* |  |
|  | | | | | | | | | | |
| Titolo :  *Title :* | **- Draft** | | | | | | | | | |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Nome & Funzione  *Name & Function* | Firma  *Signature* | Data  *Date* |  | LISTA DI DISTRIBUZIONE  *DISTRIBUTION LIST* | N | A | I |
| Preparato da:  *Prepared by:* |  |  |  | Interna / *Internal* |  |  |  |
|  |  |  |  |
| Approvato da:  *Approved by:* | Lorenzo Cibin  Massimo Vitta  Cristiano Cinquepalmi |  |  |
| Applicazione autorizzata da:  *Application authorized by:* | Lorenzo Cibin |  |  | Esterna / *External* |  |  |  |
|  |  |  |  |
| Customer / Higher Level Contractor | | | |
| Accettato da:  *Accepted by:* |  |  |  |
| Approvato da:  *Approved by:* |  |  |  |
| N=Numero di copie A=Applicazione I=Informazione  *N=Number of copy A=Application I=Information* | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Gestione documenti:  *Data Management:* |  |  |  |  |
| Firma / *Signature* Data / *Date* |  | File: | NEOSTEL-TN-CGS-001\_Coding\_Standard\_draft.doc |

| **REGISTRAZIONE DELLE MODIFICHE / *CHANGE RECORD*** | | | |
| --- | --- | --- | --- |
| EDIZIONE  *ISSUE* | DATA  *DATE* | AUTORIZZAZIONE  *CHANGE AUTHORITY* | OGGETTO DELLA MODIFICA E SEZIONI AFFETTE  *REASON FOR CHANGE AND AFFECTED SECTIONS* |
| 1 |  |  | First Issue |
|  |  |  |  |
|  |  |  |  |

| **LISTA DELLE PAGINE VALIDE / *LIST OF VALID PAGES*** | | | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| PAGINA  *PAGE* | EDIZIONE  *ISSUE* | PAGINA  *PAGE* | EDIZIONE  *ISSUE* | PAGINA  *PAGE* | EDIZIONE  *ISSUE* | PAGINA  *PAGE* | EDIZIONE  *ISSUE* | PAGINA  *PAGE* | EDIZIONE  *ISSUE* |
| 1 - | 1 |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |

**TABLE OF CONTENTS**

1. Introduction 5

1.1 Document SCOPE and Organization 6

1.2 Acronyms 6

1.3 Applicable Documents 6

1.4 Reference Documents 6

2. C++ Programming Style Guidelines 7

2.1 The C++ source-File 7

2.1.1 Use a source file header 7

2.1.2 Comment in doxygen-style 8

3. MISRA2004 compliance matrix 9

**LIST OF FIGURES**

Non è stata trovata alcuna voce dell'indice delle figure.

**LIST OF TABLES**

[Table 3‑1: NEOSTEL MISRA2004 Tailoring Chapter 1 – Environment 9](#_Toc397943724)

[Table 3‑2: NEOSTEL MISRA2004 Tailoring Chapter 2 - Language Extensions 10](#_Toc397943725)

[Table 3‑3: NEOSTEL MISRA2004 Tailoring Chapter 3 - Documentation 10](#_Toc397943726)

[Table 3‑4: NEOSTEL MISRA2004 Tailoring Chapter 4 - Documentation 11](#_Toc397943727)

[Table 3‑5: NEOSTEL MISRA2004 Tailoring Chapter 5 – Identifiers 11](#_Toc397943728)

[Table 3‑6: NEOSTEL MISRA2004 Tailoring Chapter 6 - Types 12](#_Toc397943729)

[Table 3‑7: NEOSTEL MISRA2004 Tailoring Chapter 7 - Constants 12](#_Toc397943730)

[Table 3‑8: NEOSTEL MISRA2004 Tailoring Chapter 8 - Declarations and definitions 14](#_Toc397943731)

[Table 3‑9: NEOSTEL MISRA2004 Tailoring Chapter 9 - Initialisation 14](#_Toc397943732)

[Table 3‑10: NEOSTEL MISRA2004 Tailoring Chapter 10- Arithmetic type conversions 15](#_Toc397943733)

[Table 3‑11: NEOSTEL MISRA2004 Tailoring Chapter 11 - Pointer type conversions 16](#_Toc397943734)

[Table 3‑12: NEOSTEL MISRA2004 Tailoring Chapter 12 - Expressions 18](#_Toc397943735)

[Table 3‑13: NEOSTEL MISRA2004 Tailoring Chapter 13 - Control statement expressions 18](#_Toc397943736)

[Table 3‑14: NEOSTEL MISRA2004 Tailoring Chapter 14 - Control flow 19](#_Toc397943737)

[Table 3‑15: NEOSTEL MISRA2004 Tailoring Chapter 15 - Switch statements 20](#_Toc397943738)

[Table 3‑16: NEOSTEL MISRA2004 Tailoring Chapter 16 - Functions 21](#_Toc397943739)

[Table 3‑17: NEOSTEL MISRA2004 Tailoring Chapter 17- Pointers and arrays 22](#_Toc397943740)

[Table 3‑18: NEOSTEL MISRA2004 Tailoring Chapter 18 - Structures and unions 22](#_Toc397943741)

[Table 3‑19: NEOSTEL MISRA2004 Tailoring Chapter 19 - Preprocessing and directives 24](#_Toc397943742)

[Table 3‑20: NEOSTEL MISRA2004 Tailoring Chapter 20 - Standard libraries 25](#_Toc397943743)

[Table 3‑21: NEOSTEL MISRA2004 Tailoring Chapter 21 - Run-time failures 25](#_Toc397943744)

# Introduction

## Document SCOPE and Organization

## Acronyms

|  |  |
| --- | --- |
| **AD** | Applicable Document |
| **CGS** | Compagnia Generale per lo Spazio (formerly Carlo Gavazzi Space) |
| **NEOSTEL** | NEO Survey Telescope |
|  |  |
|  |  |

## Applicable Documents

1. Statement of Work P2-NEO-V ‘NEO Survey Telescope Detailed Design’, SSA-NEO-TEL-SOW-0001, Issue 1, 17/12/2013
2. Space Situational Awareness - NEO System Requirements Document, SSA-NEO-RS-RD-0001, Issue 1, Revision 4, 05/04/2013
3. CGS Proposal “NEO Survey TELescope Design NEOSTEL”, S14-003 Is.1, April 2014

## Reference Documents

1. TELAD Design Report, TELAD-RP-CGS-001, version 1, 25/10/2011

# C++ Programming Style Guidelines

## The C++ source-File

### Use a source file header

Each source file (.cpp and .h) shall begin with a source file header in English. Use the following template with the following information at least:

/\*------------------------------------------------------------------------------------------\*/  
 @file HKDataHandler.cpp  
 @brief <purpose\_text>  
/  
/\*--------------------------------------------------------------------------------------------  
 Project: <project name>, Copyright &copy;2003-2005 OHB CGS-OHB  
 Development Tool: MS Visual C++  
 Author: <author>

Creation: yyyy-mm-dd  
 ------------------------------------------------------------------------------------------\*/

### Comment in doxygen-style

Comments of member variables and methods shall be written in ‘doxygen-style’ to use the documentation generator Doxygen.

/\*\* @brief The view for the Camera control panel. \*/

struct TableVersions  
{  
 unsigned int UnitConfVersion; /\*\*< @brief Version of the Unit Conf table. \*/  
 unsigned int HKDisplaysVersion; /\*\*< @brief Version of the HK Displays table. \*/  
};  
  
/\*\* @todo Pass "return" to child dialogs. Because they are not activated they do not get   
 the message that the return key was hit (use pretranslate message i.e.) \*/  
  
/\*\* @brief The view for the EPM control panel. \*/  
class EPMControlPanelView : public FormView  
{  
 /\*\* @brief Standard constructor. \*/  
 EPMControlPanelView();  
public:  
 /\*\* @brief Destructor  
 \*  
 \* Member dialogs must be deleted in the destructor, else there is a warning message that   
 \* OnDestroy or OnPostNcDestroy is not called of from Dialog derived classes.  
 \*/  
 virtual ~EPMControlPanelView();  
  
 UINT8 m\_CommandSource; /\*\*< @brief Source id for outgoing telecommands \*/  
  
 /\*\* @brief Check a received TM Packet  
 \*   
 \* @param[in] tmPacket The telemetry packet  
 \* @return @b true if TM packet is corrupted  
 \*/  
 bool CheckTMPacket(TmPacket\* pTMPacket);  
  
 /\*\* @brief Create a log file.  
 \*  
 \* @param[in] type Logfile type.  
 \* - @c HKLog - Start log tracer with HK log.  
 \* - @c MsgLog - Start log tracer with message log.  
 \* - @c TMLog - Not used.  
 \* - @c TCLog - Not used.  
 \*/  
 void CreateLogFile(::LogFileType type);

# MISRA2004 compliance matrix

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Environment | | | | | | |
| Rule | Description | MISRA  recommendation | applicable | PCLint level | PCLint rule | remark |
| 1.1 | All code shall conform to ISO 9899:1990 C programming language, ISO 9899, amended and corrected by ISO/IEC 9899/COR1:1995, ISO/IEC 9899/AMD1:1995, and ISO/IEC 9899/COR2: 1996 | req | yes | note | 950 |  |
| 1.2 | No reliance shall be placed on undefined or unspecified behaviour. | req | yes | - |  |  |
| 1.3 | Multiple compilers and/or languages shall only be used if there is a common defined interface standard for object code to which the languages/compilers/assemblers conform. | req | yes | - | - |  |
| 1.4 | The compiler/linker shall be checked to ensure that 31 character significance and case sensitivity are supported for external identifiers. | req | yes | info | 621 |  |
| 1.5 | Floating-point implementations should comply with a defined floating-point standard. | req |  | - | - |  |

Table 3‑1: NEOSTEL MISRA2004 Tailoring Chapter 1 – Environment

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Language extensions | | | | | | |
|  | Description | MISRA  recommendation | applicable | PCLint level | PCLint rule | remark |
| 2.1 | Assembly language shall be encapsulated and isolated. | req | yes | warning | 586 |  |
| 2.2 | Source code shall only use C-style comments. | req | yes | info | 950 |  |
| 2.3 | The character sequence /\* shall not be used within a comment. | req | yes | note | 602 |  |
| 2.4 | Sections of code should not be 'commented out'. | req | yes | - | - | manual walkthrough necessary |

Table 3‑2: NEOSTEL MISRA2004 Tailoring Chapter 2 - Language Extensions

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Documentation | | | | | | |
| Rule | Description | MISRA recommendation | applicable | PCLint level | PCLint rule | remark |
| 3.1 | All usage of implementation-defined behaviour shall be documented. | req | yes | - | - | manual walkthrough necessary |
| 3.2 | The character set and the corresponding encoding shall be documented. | req | yes | - | - | manual walkthrough necessary |
| 3.3 | The implementation of integer division in the chosen compiler should be determined, documented and taken into account. | adv | yes | - | - | manual walkthrough necessary |
| 3.4 | All uses of the #pragma directive shall be documented and explained. | req | yes | - | - | manual walkthrough necessary |
| 3.5 | If it is being relied upon, the implementation-defined behaviour and packing of bitfields shall be documented. | req | yes | - | - | manual walkthrough necessary |
| 3.6 | All libraries used in production code shall be written to comply with the provisions of this document, and shall have been subject to appropriate validation. | req | yes | - | - | covered by the PCLint Rules |

Table 3‑3: NEOSTEL MISRA2004 Tailoring Chapter 3 - Documentation

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Character sets | | | | | | |
| Rule | Description | MISRA  recommendation | applicable | PCLint level | PCLint rule | remark |
| 4.1 | Only those escape sequences that are defined in the ISO C standard shall be used. | req | yes | warning | 606 |  |
| 4.2 | Trigraphs shall not be used. | req | yes | info | 739 |  |

Table 3‑4: NEOSTEL MISRA2004 Tailoring Chapter 4 - Documentation

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Identifiers | | | | | | |
|  | Description | MISRA recommendation | applicable | PCLint level | PCLint rule | remark |
| 5.1 | Identifiers (internal and external) shall not rely on the significance of more than 31 characters. | req | yes | warning | 621 |  |
| 5.2 | Identifiers in an inner scope shall not use the same name as an identifier in an outer scope, and therefore hide that identifier. | req | yes | warning | 578 |  |
| 5.3 | A typedef name shall be a unique identifier. | req | yes | warning | 578 |  |
| warning | 623 |  |
| 5.4 | A tag name shall be a unique identifier. | req | yes | warning | 578 |  |
| error | 14 |  |
| error | 15 |  |
| 5.5 | No object or function identifier with static storage duration should be reused. | adv | yes | warning | 578 |  |
| error | 580 |  |
| 5.6 | No identifier in one name space should have the same spelling as an identifier in another name space, with the exception of structure and union member names. | adv | yes | warning | 578 |  |
| error | 580 |  |
| 5.7 | No identifier name should be reused. | adv | yes | warning | 578 |  |
| error | 580 |  |

Table 3‑5: NEOSTEL MISRA2004 Tailoring Chapter 5 – Identifiers

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Types | | | | | | |
|  | Description | MISRA recommendation | applicable | PCLint level | PCLint rule | remark |
| 6.1 | The plain char type shall be used only for the storage and use of character values. | req | yes | - | - | manual walkthrough necessary |
| 6.2 | Signed and unsigned char type shall be used only for the storage and use of numeric values. | req | yes | - | - | manual walkthrough necessary |
| 6.3 | Typedefs that indicate size and signedness should be used in place of the basic types. | adv | yes | note | 970 |  |
| 6.4 | Bit fields shall only be defined to be of type unsigned int or signed int. | req | yes | error | 46 |  |
| 6.5 | Bit fields of type signed int shall be at least 2 bits long. | req | yes | info | 806 |  |

Table 3‑6: NEOSTEL MISRA2004 Tailoring Chapter 6 - Types

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Constants | | | | | | |
|  | Description | MISRA recommendation | applicable | PCLint level | PCLint rule | remark |
| 7.1 | Octal constants (other than zero) and octal escape sequences shall not be used. | req | yes | note | 960 |  |

Table 3‑7: NEOSTEL MISRA2004 Tailoring Chapter 7 - Constants

| Declarations and definitions | | | | | | |
| --- | --- | --- | --- | --- | --- | --- |
|  | Description | MISRA recommendation | applicable | PCLint level | PCLint rule | remark |
| 8.1 | Functions shall have prototype declarations and the prototype shall be visible at both the function definition and call. | req | yes | info | 718 |  |
| info | 746 |  |
| note | 937 |  |
| note | 957 |  |
| 8.2 | Whenever an object or function is declared or defined, its type shall be explicitly stated. | req | yes | info | 745 |  |
| note | 939 |  |
| 8.3 | For each function parameter the type given in the declaration and definition shall be identical, and the return types shall also be identical. | req | yes | error | 18 |  |
| warning | 516 |  |
| warning | 532 |  |
| 8.4 | If objects or functions are declared more than once their types shall be compatible. | req | yes | error | 15 |  |
| error | 64 |
| 8.5 | There shall be no definitions of objects or functions in a header file. | req | yes | note | 960 |  |
| 8.6 | Functions shall be declared at file scope. | req | yes | note | 960- | manual walkthrough necessary |
| 8.7 | Objects shall be defined at block scope if they are only accessed from within a single function. | req | yes | - | - | manual walkthrough necessary |
| 8.8 | An external object or function shall be declared in one and only one file. | req | yes | - | - | manual walkthrough necessary |
| 8.9 | An identifier with external linkage shall have exactly one external definition. | req | yes | error | 14 |  |
| 8.10 | All declarations and definitions of objects or functions at file scope shall have internal linkage unless external linkage is required. | req | yes | warning | 765 |  |
| 8.11 | The static storage class specifier shall be used in definitions and declarations of objects and functions that have internal linkage. | req | yes | warning | 512 |  |
| 8.12 | When an array is declared with external linkage, its size shall be stated explicitly or defined implicitly by initialisation. | req | yes | error | 85 |  |

Table 3‑8: NEOSTEL MISRA2004 Tailoring Chapter 8 - Declarations and definitions

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Initialisation |  |  |  |  |  |
| Rule | Description | MISRA recommendation | applicable | PCLint level | PCLint rule | remark |
| 9.1 | Functions shall have prototype declarations and the prototype shall be visible at both the function definition and call. | req | yes | warning | 644 |  |
| info | 771 |  |
| warning | 530 |  |
| 9.2 | Braces shall be used to indicate and match the structure in the non-zero initialisation of arrays and structures. | req | yes | note | 960 |  |
| 9.3 | In an enumerator list, the '=' construct shall not be used to explicitly initialise members other than the first, unless all items are explicitly initialised. | req | yes | note | 960 |  |

Table 3‑9: NEOSTEL MISRA2004 Tailoring Chapter 9 - Initialisation

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Arithmetic type conversions | | | | | | |
|  | Description | MISRA recommendation | applicable | PCLint level | PCLint rule | remark |
| 10.1 | Functions shall have prototype declarations and the prototype shall be visible at both the function definition and call. | req | yes | warning | 524 |  |
| warning | 653 |  |
| 10.2 | The value of an expression of floating type shall not be implicitly converted to a different type if: a) it is not a conversion to a wider floating type, or b) the expression is complex, or c) the expression is a function argument, or d) the exp | req | yes | info | 747 |  |
| note | 918 |  |
| 10.3 | The value of a complex expression of integer type may only be cast to a type that is narrower and of the same signedness as the underlying type of the expression. | req | yes | note | 960 |  |
| 10.4 | The value of a complex expression of floating type may only be cast to a narrower floating type. | req | yes | note | 960 |  |
| 10.5 | If the bitwise operators ~ and << are applied to an operand of underlying type unsigned char or unsigned short, the result shall be immediately cast to the underlying type of the operand. | req | yes | info | 701 |  |
| info | 702 |  |
| 10.6 | A "U" suffix shall be applied to all constants of unsigned type. | req | yes | - | - | manual walkthrough necessary |

Table 3‑10: NEOSTEL MISRA2004 Tailoring Chapter 10- Arithmetic type conversions

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Pointer type conversions | | | | | | |
|  | Description | MISRA recommendation | applicable | PCLint level | PCLint rule | remark |
| 11.1 | Conversions shall not be performed between a pointer to a function and any type other than an integral type. | req | yes | note | 923 |  |
| 11.2 | Conversions shall not be performed between a pointer to object and any type other than an integral type, another pointer to object type or a pointer to void. | req | yes | error | 71 |  |
| 11.3 | A cast should not be performed between a pointer type and an integral type. | adv |  | note | 923 |  |
| 11.4 | A cast should not be performed between a pointer to object type and a different pointer to object type. | adv | yes | note | 926 |  |
| note | 927 |  |
| note | 928 |  |
| note | 929 |  |
| 11.5 | A cast shall not be performed that removes any const or volatile qualification from the type addressed by a pointer. | req |  |  | - | manual walkthrough necessary |

Table 3‑11: NEOSTEL MISRA2004 Tailoring Chapter 11 - Pointer type conversions

| Expressions | | | | | | |
| --- | --- | --- | --- | --- | --- | --- |
|  | Description | MISRA recommendation | applicable | PCLint level | PCLint rule | remark |
| 12.1 | Limited dependence should be placed on C's operator precedence rules in expressions. | adv | yes | info | 834 |  |
| note | 961 |  |
| 12.2 | The value of an expression shall be the same under any order of evaluation that the standard permits. | req | yes | warning | 564 |  |
| 12.3 | The sizeof operator shall not be used on expressions that contain side effects. | req | yes | note | 960 |  |
| 12.4 | The right hand operand of a logical && or || operator shall not contain side effects. | req | yes | note | 960 |  |
| 12.5 | The operands of a logical && or || shall be primary-expressions. | req | yes | - | - | manual walkthrough necessary |
| 12.6 | The operands of logical operators (&&, || and !) should be effectively Boolean. Expressions that are effectively Boolean should not be used as operands to operators other than (&&, || and !). | adv | yes | - | - | manual walkthrough necessary |
| 12.7 | Bitwise operators shall not be applied to operands whose underlying type is signed. | req | yes | note | 960 |  |
| 12.8 | 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. | req | yes | warning | 572 |  |
| 12.9 | The unary minus operator shall not be applied to an expression whose underlying type is unsigned. | req | yes | warning | 501 |  |
| 12.10 | The comma operator shall not be used. | req | yes | note | 960 |  |
| 12.11 | Evaluation of constant unsigned integer expressions should not lead to wrap-around. | adv | yes | warning | 648 |  |
| 12.12 | The underlying bit representations of floating-point values shall not be used. | req | yes | - | - | manual walkthrough necessary |
| 12.13 | The increment (++) and decrement (--) operators should not be mixed with other operators in an expression. | adv | no | - | - | manual walkthrough necessary |

Table 3‑12: NEOSTEL MISRA2004 Tailoring Chapter 12 - Expressions

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Control statement expressions | | | | | | |
| Rule | Description | MISRA recommendation | applicable | PCLint level | PCLint rule | remark |
| 13.1 | Assignment operators shall not be used in expressions that yield a Boolean value. | req | yes | info | 720 |  |
| error | 82 |  |
| 13.2 | Tests of a value against zero should be made explicit, unless the operand is effectively Boolean. | adv | yes | info | 720 |  |
| 13.3 | Floating-point expressions shall not be tested for equality or inequality. | req | yes | info | 777 |  |
| 13.4 | The controlling expression of a for statement shall not contain any objects of floating type. | req | yes | note | 960 |  |
| 13.5 | The three expressions of a for statement shall be concerned only with loop control. | req | yes | - | - | manual walkthrough necessary |
| 13.6 | Numeric variables being used within a for loop for iteration counting shall not be modified in the body of the loop. | req | yes | - | - | manual walkthrough necessary |
| 13.7 | Boolean operations whose results are invariant shall not be permitted. | req | yes | warning | 506 |  |

Table 3‑13: NEOSTEL MISRA2004 Tailoring Chapter 13 - Control statement expressions

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Control flow | | | | | | |
| Rule | Description | MISRA recommendation | applicable | PCLint level | PCLint rule | remark |
| 14.1 | There shall be no unreachable code. | req | yes | warning | 506 |  |
| warning | 527 |  |
| warning | 681 |  |
| info | 827 |  |
| 14.2 | All non-null statements shall either (i) have at least one side-effect however executed, or (ii) cause control flow to change. | req | yes | warning | 505 |  |
| warning | 522 |  |
| 14.3 | Before preprocessing, a null statement shall only occur on a line by itself; it may be followed by a comment provided that the first character following the null statement is a white-space character. | req | yes | note | 960 |  |
| 14.4 | The goto statement shall not be used. | req | yes | info | 801 |  |
| 14.5 | The continue statement shall not be used. | req | yes | note | 960 |  |
| 14.6 | For any iteration statement there shall be at most one break statement used for loop termination. | req | yes | note | 960 |  |
| 14.7 | A function shall have a single point of exit at the end of the function. | req | yes | note | 904 |  |
| 14.8 | The statement forming the body of a switch, while, do ... while or for statement shall be a compound statement. | req | yes | note | 960 |  |
| 14.9 | An if (expression) construct shall be followed by a compound statement. The else keyword shall be followed by either a compound statement, or another if statement. | req | yes | note | 960 |  |
| 14.10 | All if ... else if constructs shall be terminated with an else clause. | req | yes | note | 960 |  |

Table 3‑14: NEOSTEL MISRA2004 Tailoring Chapter 14 - Control flow

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Switch statements | | | | | | |
| Rule | Description | MISRA recommendation | applicable | PCLint level | PCLint rule | remark |
| 15.0 | The MISRA C switch syntax shall be used | req | yes | - | - | manual walkthrough necessary |
| 15.1 | A switch label shall only be used when the most closely-enclosing compound statement is the body of a switch statement. | req | yes | error | 44 |  |
| 15.2 | An unconditional break statement shall terminate every non-empty switch clause. | req | yes | warning | 616 |  |
| 15.3 | The final clause of a switch statement shall be the default clause. | req | yes | info | 744 |  |
| 15.4 | A switch expression shall not represent a value that is effectively Boolean. | req | yes | note | 960 |  |
| 15.5 | Every switch statement shall have at least one case clause. | req | yes | info | 764 |  |

Table 3‑15: NEOSTEL MISRA2004 Tailoring Chapter 15 - Switch statements

| Functions | | | | | | |
| --- | --- | --- | --- | --- | --- | --- |
| Rule | Description | MISRA recommendation | applicable | PCLint level | PCLint rule | remark |
| 16.1 | Functions shall not be defined with a variable number of arguments. | req | yes | note | 960 |  |
| 16.2 | Functions shall not call themselves, either directly or indirectly. | req | yes | note | 974 |  |
| 16.3 | Identifiers shall be given for all of the parameters in a function prototype declaration. | req | yes | note | 960 |  |
| 16.4 | The identifiers used in the declaration and definition of a function shall be identical. | req | yes | - | - | manual walkthrough necessary |
| 16.5 | Functions with no parameters shall be declared with parameter type void. | req | yes | note | 937 |  |
| 16.6 | The number of arguments passed to a function shall match the number of parameters. | req | yes | error | 118 |  |
| error | 119 |  |
| 16.7 | A pointer parameter in a function prototype should be declared as pointer to const if the pointer is not used to modify the addressed object. | adv | yes | info | 818 |  |
| 16.8 | All exit paths from a function with non-void return type shall have an explicit return statement with an expression. | req | yes | warning | 533 |  |
| 16.9 | A function identifier shall only be used with either a preceding &, or with a parenthesised parameter list, which may be empty. | req | yes | - | - | manual walkthrough necessary |
| 16.10 | If a function returns error information, then that error information shall be tested. | req | yes | warning | 534 |  |

Table 3‑16: NEOSTEL MISRA2004 Tailoring Chapter 16 - Functions

| Pointer and arrays | | | | | | |
| --- | --- | --- | --- | --- | --- | --- |
| Rule | Description | MISRA recommendation | applicable | PCLint level | PCLint rule | remark |
| 17.1 | Pointer arithmetic shall only be applied to pointers that address an array or array element. | req | yes | - | - | manual walkthrough necessary |
| 17.2 | Pointer subtraction shall only be applied to pointers that address elements of the same array. | req | yes | note | 946 |  |
| note | 947 |  |
| 17.3 | >, >=, <, <= shall not be applied to pointer types except where they point to the same array. | req | yes | note | see rule 17.2 |  |
| 17.4 | Array indexing shall be the only allowed form of pointer arithmetic. | req | yes | - | - | manual walkthrough necessary |
| 17.5 | The declaration of objects should contain no more than 2 levels of pointer indirection. | adv | yes | note | 961 |  |
| 17.6 | The address of an object with automatic storage shall not be assigned to another object that may persist after the first object has ceased to exist. | req | yes | info | 733 |  |
| info | 789 |  |

Table 3‑17: NEOSTEL MISRA2004 Tailoring Chapter 17- Pointers and arrays

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Structures and unions | | | | | | |
| Rule | Description | MISRA recommendation | applicable | PCLint level | PCLint rule | remark |
| 18.1 | All structure and union types shall be complete at the end of a translation unit. | req | yes | error | 43 |  |
| 18.2 | An object shall not be assigned to an overlapping object. | req | yes | - | - | manual walkthrough necessary |
| 18.3 | An area of memory shall not be reused for unrelated purposes. | req | yes | - | - | manual walkthrough necessary |
| 18.4 | Unions shall not be used. | req |  | note | 960 |  |

Table 3‑18: NEOSTEL MISRA2004 Tailoring Chapter 18 - Structures and unions

| Preprocessing directives | | | | | | |
| --- | --- | --- | --- | --- | --- | --- |
| Rule | Description | MISRA recommendation | applicable | PCLint level | PCLint rule | remark |
| 19.1 | #include statements in a file should only be preceded by other preprocessor directives or comments. | adv | yes | note | 961 |  |
| 19.2 | Non-standard characters should not occur in header file names in #include directives. | adv | yes | note | 961 |  |
| 19.3 | The #include directive shall be followed by either a <filename> or "filename" sequence. | req | yes | error | 12 |  |
| 19.4 | C macros shall only expand to a braced initialiser, a constant, a parenthesised expression, a type qualifier, a storage class specifier, or a do-while-zero construct. | req | yes | info | 773 |  |
| 19.5 | Macros shall not be #define'd or #undef'd within a block. | req | yes | note | 960 |  |
| 19.6 | #undef shall not be used. | req | yes | note | 961 |  |
| 19.7 | A function should be used in preference to a function-like macro. | adv | yes | note | 961 |  |
| 19.8 | A function-like macro shall not be invoked without all of its arguments. | req | yes | error | 131 |  |
| 19.9 | Arguments to a function-like macro shall not contain tokens that look like preprocessing directives. | req | yes | warning | 436 |  |
| 19.10 | In the definition of a function-like macro each instance of a parameter shall be enclosed in parentheses unless it is used as the operand of # or ##. | req | yes | note | see rule 19.4 |  |
| 19.11 | All macro identifiers in preprocessor directives shall be defined before use, except in #ifdef and #ifndef preprocessor directives and the defined() operator. | req | yes | warning | 553 |  |
| 19.12 | There shall be at most one occurrence of the # or ## preprocessor operators in a single macro definition. | req | yes | note | 960 |  |
| 19.13 | The # and ## preprocessor operators should not be used. | adv | yes | note | 961 |  |
| 19.14 | The defined preprocessor operator shall only be used in one of the two standard forms. | req | yes | note | 960 |  |
| 19.15 | Precautions shall be taken in order to prevent the contents of a header file being included twice. | req | yes | warning | 537 |  |
| 19.16 | Preprocessing directives shall be syntactically meaningful even when excluded by the preprocessor. | req | yes | - | - | manual walkthrough necessary |
| 19.17 | All #else, #elif and #endif preprocessor directives shall reside in the same file as the #if or #ifdef directive to which they are related. | req | yes | warning | 405 |  |

Table 3‑19: NEOSTEL MISRA2004 Tailoring Chapter 19 - Preprocessing and directives

| Standard libraries | | | | | | |
| --- | --- | --- | --- | --- | --- | --- |
| Rule | Description | MISRA recommendation | applicable | PCLint level | PCLint rule | remark |
| 20.1 | Reserved identifiers, macros and functions in the standard library, shall not be defined, redefined or undefined. | req | yes | warning | 683 |  |
| 20.2 | The names of standard library macros, objects and functions shall not be reused. | req | yes | - | - | manual walkthrough necessary |
| 20.3 | The validity of values passed to library functions shall be checked. | req | no | - | 100 calls are monitored |  |
| 20.4 | Dynamic heap memory allocation shall not be used. | req | yes | - | 586 |  |
| 20.5 | The error indicator errno shall not be used. | req | yes | - | deprecate |  |
| 20.6 | The macro offsetof, in library <stddef.h>, shall not be used. | req | yes | note | deprecate |  |
| 20.7 | The setjmp macro and the longjmp function shall not be used. | req | yes | note | deprecate |  |
| 20.8 | The signal handling facilities of <signal.h> shall not be used. | req |  |  | 586 |  |
| 20.9 | The input/output library <stdio.h> shall not be used in production code. | req | yes | info | 829 |  |
| 20.10 | The library functions atof, atoi and atol from library <stdlib.h> shall not be used. | req | yes | warning | 586 |  |
| 20.11 | The library functions abort, exit, getenv and system from library <stdlib.h> shall not be used. | req | yes | warning | 586 |  |
| 20.12 | The time handling functions of library <time.h> shall not be used. | req | yes | warning | 586 |  |

Table 3‑20: NEOSTEL MISRA2004 Tailoring Chapter 20 - Standard libraries

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Run-time failures | | | | | | |
| Rule | Description | MISRA recommendation | applicable | PCLint level | PCLint rule | remark |
| 21.1 | Minimisation of run-time failures shall be ensured by the use of at least one of a) static analysis tools/techniques, b) dynamic analysis/techniques, c) explicit coding checks to handle run-time faults | req | yes | - | - | achieved by use of Lint |

Table 3‑21: NEOSTEL MISRA2004 Tailoring Chapter 21 - Run-time failures