Description Resource Path Location Type

"../main.c", line 75: Assembly statement "NOP" creates a label, which may not be what was intended. Use a colon after a label or a space before a non-label to silence the warning. "../main.c", line 75: Assembly statement "NOP" creates a label, which may not be what was intended. Use a colon after a label or a space before a non-label to silence the warning. main.c /test C/C++ Problem

"../src/HAL/keypad/keypad_program.c", line 27: Assembly statement "NOP" creates a label, which may not be what was intended. Use a colon after a label or a space before a non-label to silence the warning. "../src/HAL/keypad/keypad_program.c", line 27: Assembly statement "NOP" creates a label, which may not be what was intended. Use a colon after a label or a space before a non-label to silence the warning. keypad_program.c /test/src/HAL/keypad C/C++ Problem

"../src/HAL/LCD/LCD_Program.c", line 16: Assembly statement "NOP" creates a label, which may not be what was intended. Use a colon after a label or a space before a non-label to silence the warning. "../src/HAL/LCD/LCD_Program.c", line 16: Assembly statement "NOP" creates a label, which may not be what was intended. Use a colon after a label or a space before a non-label to silence the warning. LCD_Program.c /test/src/HAL/LCD C/C++ Problem

(MISRA-C:2004 19.4/R) C macros shall only expand to a braced initialiser, a constant, a string literal, a parenthesised expression, a type qualifier, a storage class specifier, or a do-while-zero construct: ccs=ccs (MISRA-C:2004 19.4/R) C macros shall only expand to a braced initialiser, a constant, a string literal, a parenthesised expression, a type qualifier, a storage class specifier, or a do-while-zero construct: ccs=ccs GPIO.c /test/src/MCAL/GPIO C/C++ Problem

(MISRA-C:2004 19.4/R) C macros shall only expand to a braced initialiser, a constant, a string literal, a parenthesised expression, a type qualifier, a storage class specifier, or a do-while-zero construct: ccs=ccs (MISRA-C:2004 19.4/R) C macros shall only expand to a braced initialiser, a constant, a string literal, a parenthesised expression, a type qualifier, a storage class specifier, or a do-while-zero construct: ccs=ccs GPIO_config.c /test/src/MCAL/GPIO C/C++ Problem

(MISRA-C:2004 19.4/R) C macros shall only expand to a braced initialiser, a constant, a string literal, a parenthesised expression, a type qualifier, a storage class specifier, or a do-while-zero construct: ccs=ccs (MISRA-C:2004 19.4/R) C macros shall only expand to a braced initialiser, a constant, a string literal, a parenthesised expression, a type qualifier, a storage class specifier, or a do-while-zero construct: ccs=ccs GPT.c /test/src/MCAL/GPT C/C++ Problem

(MISRA-C:2004 19.4/R) C macros shall only expand to a braced initialiser, a constant, a string literal, a parenthesised expression, a type qualifier, a storage class specifier, or a do-while-zero construct: ccs=ccs (MISRA-C:2004 19.4/R) C macros shall only expand to a braced initialiser, a constant, a string literal, a parenthesised expression, a type qualifier, a storage class specifier, or a do-while-zero construct: ccs=ccs GPT_config.c /test/src/MCAL/GPT C/C++ Problem

(MISRA-C:2004 19.4/R) C macros shall only expand to a braced initialiser, a constant, a string literal, a parenthesised expression, a type qualifier, a storage class specifier, or a do-while-zero construct: ccs=ccs (MISRA-C:2004 19.4/R) C macros shall only expand to a braced initialiser, a constant, a string literal, a parenthesised expression, a type qualifier, a storage class specifier, or a do-while-zero construct: ccs=ccs LCD_Program.c /test/src/HAL/LCD C/C++ Problem

(MISRA-C:2004 19.4/R) C macros shall only expand to a braced initialiser, a constant, a string literal, a parenthesised expression, a type qualifier, a storage class specifier, or a do-while-zero construct: ccs=ccs
(MISRA-C:2004 19.4/R) C macros shall only expand to a braced initialiser, a constant, a string literal, a parenthesised expression, a type qualifier, a storage class specifier, or a do-while-zero construct: ccs=ccs
SCB.c /test/src/MCAL/SCB C/C++ Problem

(MISRA-C:2004 19.4/R) C macros shall only expand to a braced initialiser, a constant, a string literal, a parenthesised expression, a type qualifier, a storage class specifier, or a do-while-zero construct: ccs=ccs
(MISRA-C:2004 19.4/R) C macros shall only expand to a braced initialiser, a constant, a string literal, a parenthesised expression, a type qualifier, a storage class specifier, or a do-while-zero construct: ccs=ccs
SCB_config.c /test/src/MCAL/SCB C/C++ Problem

(MISRA-C:2004 19.4/R) C macros shall only expand to a braced initialiser, a constant, a string literal, a parenthesised expression, a type qualifier, a storage class specifier, or a do-while-zero construct: ccs=ccs
(MISRA-C:2004 19.4/R) C macros shall only expand to a braced initialiser, a constant, a string literal, a parenthesised expression, a type qualifier, a storage class specifier, or a do-while-zero construct: ccs=ccs
SYSTICK_Program.c /test/src/MCAL/SYSTICK C/C++ Problem

(MISRA-C:2004 19.4/R) C macros shall only expand to a braced initialiser, a constant, a string literal, a parenthesised expression, a type qualifier, a storage class specifier, or a do-while-zero construct: ccs=ccs
(MISRA-C:2004 19.4/R) C macros shall only expand to a braced initialiser, a constant, a string literal, a parenthesised expression, a type qualifier, a storage class specifier, or a do-while-zero construct: ccs=ccs
keypad_program.c /test/src/HAL/keypad C/C++ Problem

(MISRA-C:2004 19.4/R) C macros shall only expand to a braced initialiser, a constant, a string literal, a parenthesised expression, a type qualifier, a storage class specifier, or a do-while-zero construct: ccs=ccs
(MISRA-C:2004 19.4/R) C macros shall only expand to a braced initialiser, a constant, a string literal, a parenthesised expression, a type qualifier, a storage class specifier, or a do-while-zero construct: ccs=ccs
main.c /test C/C++ Problem

(MISRA-C:2004 19.4/R) C macros shall only expand to a braced initialiser, a constant, a string literal, a parenthesised expression, a type qualifier, a storage class specifier, or a do-while-zero construct: ccs=ccs
(MISRA-C:2004 19.4/R) C macros shall only expand to a braced initialiser, a constant, a string literal, a parenthesised expression, a type qualifier, a storage class specifier, or a do-while-zero construct: ccs=ccs
tm4c123gh6pm_startup_ccs.c /test C/C++ Problem

#1-D last line of file ends without a newline SYSTICK_Interface.h /test/inc/MCAL/SYSTICK line 16 C/C++ Problem

#112-D statement is unreachable main.c /test line 154C/C++ Problem

#1376-D (MISRA-C:2004 1.1/R) Ensure strict ANSI C mode (-ps) is enabled .ccsproject /test line 42, external location: C:\ti\ccsv7\tools\compiler\ti-cgt-arm_16.9.4.LTS\include\stdint.h C/C++ Problem

#1376-D (MISRA-C:2004 1.1/R) Ensure strict ANSI C mode (-ps) is enabled STD_TYPES.h /test/LIB line 13 C/C++ Problem

#1383-D (MISRA-C:2004 8.1/R) Functions shall have prototype declarations and the prototype shall be visible at both the function definition and call GPT.c /test/src/MCAL/GPT line 16 C/C++ Problem

#1383-D (MISRA-C:2004 8.1/R) Functions shall have prototype declarations and the prototype shall be visible at both the function definition and call GPT.c /test/src/MCAL/GPT line 151C/C++ Problem

#1383-D (MISRA-C:2004 8.1/R) Functions shall have prototype declarations and the prototype shall be visible at both the function definition and call LCD_Program.c /test/src/HAL/LCD line 11 C/C++ Problem

#1383-D (MISRA-C:2004 8.1/R) Functions shall have prototype declarations and the prototype shall be visible at both the function definition and call SYSTICK_Program.c /test/src/MCAL/SYSTICKline 99 C/C++ Problem

#1383-D (MISRA-C:2004 8.1/R) Functions shall have prototype declarations and the prototype shall be visible at both the function definition and call keypad_program.c /test/src/HAL/keypad line 22 C/C++ Problem

#1383-D (MISRA-C:2004 8.1/R) Functions shall have prototype declarations and the prototype shall be visible at both the function definition and call main.c /test line 40 C/C++ Problem

#1383-D (MISRA-C:2004 8.1/R) Functions shall have prototype declarations and the prototype shall be visible at both the function definition and call main.c /test line 70 C/C++ Problem

#1383-D (MISRA-C:2004 8.1/R) Functions shall have prototype declarations and the prototype shall be visible at both the function definition and call main.c /test line 83 C/C++ Problem

#1384-D (MISRA-C:2004 8.1/R) Functions shall have prototype declarations and the prototype shall be visible at both the function definition and call ("lcd_void_send_char") keypad_program.c /test/src/HAL/keypad line 98 C/C++ Problem

#1384-D (MISRA-C:2004 8.1/R) Functions shall have prototype declarations and the prototype shall be visible at both the function definition and call ("lcd_void_send_command") keypad_program.c /test/src/HAL/keypad line 96 C/C++ Problem

#1384-D (MISRA-C:2004 8.1/R) Functions shall have prototype declarations and the prototype shall be visible at both the function definition and call ("READ_BIT") SYSTICK_Program.c /test/src/MCAL/SYSTICKline 54 C/C++ Problem

#1384-D (MISRA-C:2004 8.1/R) Functions shall have prototype declarations and the prototype shall be visible at both the function definition and call ("READ_BIT") SYSTICK_Program.c /test/src/MCAL/SYSTICKline 109C/C++ Problem

#1386-D (MISRA-C:2004 8.6/R) Functions shall be declared at file scope (function "lcd_void_send_char") keypad_program.c /test/src/HAL/keypad line 98 C/C++ Problem

#1386-D (MISRA-C:2004 8.6/R) Functions shall be declared at file scope (function "lcd_void_send_command") keypad_program.c /test/src/HAL/keypad line 96 C/C++ Problem

#1386-D (MISRA-C:2004 8.6/R) Functions shall be declared at file scope (function "READ_BIT") SYSTICK_Program.c /test/src/MCAL/SYSTICKline 54 C/C++ Problem

#1386-D (MISRA-C:2004 8.6/R) Functions shall be declared at file scope (function "READ_BIT") SYSTICK_Program.c /test/src/MCAL/SYSTICKline 109C/C++ Problem

#1389-D (MISRA-C:2004 8.12/R) When an array is declared with external linkage, its size shall be stated explicitly or defined implicitly by initialisation SCB_interface.h /test/inc/MCAL/SCB line 30 C/C++ Problem

#1391-D (MISRA-C:2004 9.2/R) Braces shall be used to indicate and match the structure in the non-zero initialisation of arrays and structures GPIO_config.c /test/src/MCAL/GPIO line 56 C/C++ Problem

#1392-D (MISRA-C:2004 9.3/R) In an enumerator list, the '=' construct shall not be used to explicitly initialise members other than the first, unless all items are explicitly initialised GPIO_interface.h /test/inc/MCAL/GPIO line 81 C/C++ Problem

#1393-D (MISRA-C:2004 10.1/R) The value of an expression of integer type shall not be implicitly converted to a different underlying type if it is not a conversion to a wider integer type of the same signedness GPIO.c /test/src/MCAL/GPIO line 13 C/C++ Problem

#1393-D (MISRA-C:2004 10.1/R) The value of an expression of integer type shall not be implicitly converted to a different underlying type if it is not a conversion to a wider integer type of the same signedness GPIO.c /test/src/MCAL/GPIO line 16 C/C++ Problem

#1393-D (MISRA-C:2004 10.1/R) The value of an expression of integer type shall not be implicitly converted to a different underlying type if it is not a conversion to a wider integer type of the same signedness GPIO.c /test/src/MCAL/GPIO line 82 C/C++ Problem

#1393-D (MISRA-C:2004 10.1/R) The value of an expression of integer type shall not be implicitly converted to a different underlying type if it is not a conversion to a wider integer type of the same signedness GPIO.c /test/src/MCAL/GPIO line 89 C/C++ Problem

#1393-D (MISRA-C:2004 10.1/R) The value of an expression of integer type shall not be implicitly converted to a different underlying type if it is not a conversion to a wider integer type of the same signedness GPIO.c /test/src/MCAL/GPIO line 96 C/C++ Problem

#1393-D (MISRA-C:2004 10.1/R) The value of an expression of integer type shall not be implicitly converted to a different underlying type if it is not a conversion to a wider integer type of the same signedness GPIO.c /test/src/MCAL/GPIO line 104C/C++ Problem

#1393-D (MISRA-C:2004 10.1/R) The value of an expression of integer type shall not be implicitly converted to a different underlying type if it is not a conversion to a wider integer type of the same signedness GPIO.c /test/src/MCAL/GPIO line 112C/C++ Problem

#1393-D (MISRA-C:2004 10.1/R) The value of an expression of integer type shall not be implicitly converted to a different underlying type if it is not a conversion to a wider integer type of the same signedness GPIO.c /test/src/MCAL/GPIO line 119C/C++ Problem

#1393-D (MISRA-C:2004 10.1/R) The value of an expression of integer type shall not be implicitly converted to a different underlying type if it is not a conversion to a wider integer type of the same signedness GPIO.c /test/src/MCAL/GPIO line 131C/C++ Problem

#1393-D (MISRA-C:2004 10.1/R) The value of an expression of integer type shall not be implicitly converted to a different underlying type if it is not a conversion to a wider integer type of the same signedness GPIO.c /test/src/MCAL/GPIO line 138C/C++ Problem

#1393-D (MISRA-C:2004 10.1/R) The value of an expression of integer type shall not be implicitly converted to a different underlying type if it is not a conversion to a wider integer type of the same signedness GPIO.c /test/src/MCAL/GPIO line 145C/C++ Problem

#1393-D (MISRA-C:2004 10.1/R) The value of an expression of integer type shall not be implicitly converted to a different underlying type if it is not a conversion to a wider integer type of the same signedness GPIO.c /test/src/MCAL/GPIO line 153C/C++ Problem

#1393-D (MISRA-C:2004 10.1/R) The value of an expression of integer type shall not be implicitly converted to a different underlying type if it is not a conversion to a wider integer type of the same signedness GPIO.c /test/src/MCAL/GPIO line 161C/C++ Problem

#1393-D (MISRA-C:2004 10.1/R) The value of an expression of integer type shall not be implicitly converted to a different underlying type if it is not a conversion to a wider integer type of the same signedness GPIO.c /test/src/MCAL/GPIO line 169C/C++ Problem

#1393-D (MISRA-C:2004 10.1/R) The value of an expression of integer type shall not be implicitly converted to a different underlying type if it is not a conversion to a wider integer type of the same signedness GPIO.c /test/src/MCAL/GPIO line 181C/C++ Problem

#1393-D (MISRA-C:2004 10.1/R) The value of an expression of integer type shall not be implicitly converted to a different underlying type if it is not a conversion to a wider integer type of the same signedness GPIO.c /test/src/MCAL/GPIO line 188C/C++ Problem

#1393-D (MISRA-C:2004 10.1/R) The value of an expression of integer type shall not be implicitly converted to a different underlying type if it is not a conversion to a wider integer type of the same signedness GPIO.c /test/src/MCAL/GPIO line 195C/C++ Problem

#1393-D (MISRA-C:2004 10.1/R) The value of an expression of integer type shall not be implicitly converted to a different underlying type if it is not a conversion to a wider integer type of the same signedness GPIO.c /test/src/MCAL/GPIO line 202C/C++ Problem

#1393-D (MISRA-C:2004 10.1/R) The value of an expression of integer type shall not be implicitly converted to a different underlying type if it is not a conversion to a wider integer type of the same signedness GPIO.c /test/src/MCAL/GPIO line 210C/C++ Problem

#1393-D (MISRA-C:2004 10.1/R) The value of an expression of integer type shall not be implicitly converted to a different underlying type if it is not a conversion to a wider integer type of the same signedness GPIO.c /test/src/MCAL/GPIO line 218C/C++ Problem

#1393-D (MISRA-C:2004 10.1/R) The value of an expression of integer type shall not be implicitly converted to a different underlying type if it is not a conversion to a wider integer type of the same signedness GPIO.c /test/src/MCAL/GPIO line 243C/C++ Problem

#1393-D (MISRA-C:2004 10.1/R) The value of an expression of integer type shall not be implicitly converted to a different underlying type if it is not a conversion to a wider integer type of the same signedness GPIO.c /test/src/MCAL/GPIO line 246C/C++ Problem

#1393-D (MISRA-C:2004 10.1/R) The value of an expression of integer type shall not be implicitly converted to a different underlying type if it is not a conversion to a wider integer type of the same signedness GPIO.c /test/src/MCAL/GPIO line 249C/C++ Problem

#1393-D (MISRA-C:2004 10.1/R) The value of an expression of integer type shall not be implicitly converted to a different underlying type if it is not a conversion to a wider integer type of the same signedness GPIO.c /test/src/MCAL/GPIO line 252C/C++ Problem

#1393-D (MISRA-C:2004 10.1/R) The value of an expression of integer type shall not be implicitly converted to a different underlying type if it is not a conversion to a wider integer type of the same signedness GPIO.c /test/src/MCAL/GPIO line 255C/C++ Problem

#1393-D (MISRA-C:2004 10.1/R) The value of an expression of integer type shall not be implicitly converted to a different underlying type if it is not a conversion to a wider integer type of the same signedness GPIO.c /test/src/MCAL/GPIO line 258C/C++ Problem

#1393-D (MISRA-C:2004 10.1/R) The value of an expression of integer type shall not be implicitly converted to a different underlying type if it is not a conversion to a wider integer type of the same signedness GPIO.c /test/src/MCAL/GPIO line 266C/C++ Problem

#1393-D (MISRA-C:2004 10.1/R) The value of an expression of integer type shall not be implicitly converted to a different underlying type if it is not a conversion to a wider integer type of the same signedness GPIO.c /test/src/MCAL/GPIO line 267C/C++ Problem

#1393-D (MISRA-C:2004 10.1/R) The value of an expression of integer type shall not be implicitly converted to a different underlying type if it is not a conversion to a wider integer type of the same signedness GPIO.c /test/src/MCAL/GPIO line 272C/C++ Problem

#1393-D (MISRA-C:2004 10.1/R) The value of an expression of integer type shall not be implicitly converted to a different underlying type if it is not a conversion to a wider integer type of the same signedness GPIO.c /test/src/MCAL/GPIO line 278C/C++ Problem

#1393-D (MISRA-C:2004 10.1/R) The value of an expression of integer type shall not be implicitly converted to a different underlying type if it is not a conversion to a wider integer type of the same signedness GPIO.c /test/src/MCAL/GPIO line 284C/C++ Problem

#1393-D (MISRA-C:2004 10.1/R) The value of an expression of integer type shall not be implicitly converted to a different underlying type if it is not a conversion to a wider integer type of the same signedness GPIO.c /test/src/MCAL/GPIO line 290C/C++ Problem

#1393-D (MISRA-C:2004 10.1/R) The value of an expression of integer type shall not be implicitly converted to a different underlying type if it is not a conversion to a wider integer type of the same signedness GPIO.c /test/src/MCAL/GPIO line 296C/C++ Problem