

ISO/IEC TR 18037:2008-06 (E)

Programming languages - C - Extensions to support embedded processors

Contents		Page
1	SCOPE	1
2	NORMATIVE REFERENCES	1
3	CONFORMANCE	1
4	FIXED-POINT ARITHMETIC	2
4.1	Overview and principles of the fixed-point data types	2
4.1.1	The data types	2
4.1.2	Spelling of the new keywords	3
4.1.3	Rounding and Overflow	4
4.1.4	Type conversion, usual arithmetic conversions	5
4.1.5	Fixed-point constants	6
4.1.6	Operations involving fixed-point types	7
4.1.7	Fixed-point functions	9
4.1.8	Fixed-point definitions <stdfix.h>	11
4.1.9	Formatted I/O functions for fixed-point arguments	11
4.2	Detailed changes to ISO/IEC 9899:1999	12
5	NAMED ADDRESS SPACES AND NAMED-REGISTER STORAGE CLASSES	37
5.1	Overview and principles of named address spaces	37
5.1.1	Additional address spaces	37
5.1.2	Address-space type qualifiers	37
5.1.3	Address space nesting and rules for pointers	38
5.1.4	Standard library support	39
5.2	Overview and principles of named-register storage classes	39
5.2.1	Access to machine registers	39
5.2.2	Named-register storage-class specifiers	39
5.2.3	Ensuring correct side effects via objects allocated in registers	41
5.2.4	Relationship between named registers and I/O-register designators	41
5.3	Detailed changes to ISO/IEC 9899:1999	41
6	BASIC I/O HARDWARE ADDRESSING	49
6.1	Rationale	49
6.1.1	Basic Standardization Objectives	49
6.2	Terminology	49
6.3	Basic I/O Hardware addressing header <iohw.h>	51
6.3.1	Standardization principles	51
6.3.2	The abstract model	52
6.4	Specifying I/O registers	54
6.4.1	I/O-register designators	54
6.4.2	Accesses to individual I/O registers	54
6.4.3	I/O register buffers	55
6.4.4	I/O groups	56
6.4.5	Direct and indirect designators	56
6.4.6	Operations on I/O groups	57
6.5	Detailed changes to ISO/IEC 9899:1999	58
ANNEX A - FIXED-POINT ARITHMETIC		65
A.1	Fixed-point datatypes	65
A.1.1	Introduction	65

A.2	Number of data bits in <code>_Fract</code> versus <code>_Accum</code>	68
A.3	Possible Data Type Implementations	69
A.4	Rounding and Overflow	70
A.5	Type conversions, usual arithmetic conversions	71
A.6	Operations involving fixed-point types	71
A.7	Exception for 1 and -1 Multiplication Results	72
A.8	Linguistic Variables and unsigned <code>_Fract</code> : an example of unsigned fixed-point	73
ANNEX B - NAMED ADDRESS SPACES AND NAMED-REGISTER STORAGE CLASSES		74
B.1	Embedded systems extended memory support	74
B.1.1	Modifiers for named address spaces	74
B.1.2	Application-defined multiple address space support	75
B.1.3	I/O register definition for intrinsic or user defined address spaces	76
ANNEX C - IMPLEMENTING THE <IOHW.H> HEADER		78
C.1	General	78
C.1.1	Recommended steps	78
C.1.2	Compiler considerations	78
C.2	Overview of I/O Hardware Connection Options	79
C.2.1	Multi-Addressing and I/O Register Endianness	79
C.2.2	Address Interleaving	80
C.2.3	I/O Connection Overview:	81
C.2.4	Generic buffer index	81
C.3	I/O-register designators for different I/O addressing methods	82
C.4	Atomic operation	83
C.5	Read-modify-write operations and multi-addressing cases	83
C.6	I/O initialization	84
C.7	Intrinsic Features for I/O Hardware Access	85
ANNEX D - MIGRATION PATH FOR <IOHW.H> IMPLEMENTATIONS		86
D.1	Migration path for <code><iohw.h></code> implementations	86
D.2	<code><iohw.h></code> implementation based on C macros	86
D.2.1	The access specification method	86
D.2.2	An <code><iohw.h></code> implementation technique	87
D.2.3	Features	87
D.2.4	The <code><iohw.h></code> header	88
D.2.5	The user's I/O-register designator definitions	91
D.2.6	The driver function	92
ANNEX E - FUNCTIONALITY NOT INCLUDED IN THIS TECHNICAL REPORT		93
E.1	Circular buffers	93
E.2	Complex data types	94
E.3	Consideration of BCD data types for Embedded Systems	94
E.4	Modwrap overflow	94
ANNEX F - C++ COMPATIBILITY AND MIGRATION ISSUES		96
F.1	Fixed-point Arithmetic	96
F.2	Multiple Address Spaces Support	96
F.3	Basic I/O Hardware Addressing	96
ANNEX G - UPDATES AND CHANGES IN THE SECOND EDITION OF TR 18037		97