|  |  |
| --- | --- |
|  |  |
|  |  |

T Engineering Naming Standard

**Table of Contents**

1. Introduction 4

1.1 Document Overview 4

1.2 Applicability 4

1.3 Assumptions 4

2. Naming Standard 5

2.1 Naming Fields 5

2.1.1 *SW-unit* 5

2.1.2 *storage-type* 5

2.1.3 *data-type* 8

2.1.4 *measurement-quantity* 8

2.1.5 *descriptive-abbreviation* 13

2.1.5.1 Words 13

2.1.5.2 Abbreviations 13

2.1.5.3 Acronyms 13

2.1.5.4 Numbers 13

**Table of Tables**

Table 1: Naming Overview 5

Table 2: NVM Based Values for the *storage-type* Field 7

Table 4: Type Based Values for the *storage-type* Field 7

Table 5: Values for the *data-type* Field 8

Table 6: ISO Values for the *measurement-quantity* Field for Continuous Data 10

Table 8: Values for the *measurement-quantity* Field for Derivatives of Continuous Data 12

Table 9: Values for the *measurement-quantity* Field for Discrete Data 12

Table 10: Values for the *measurement-quantity* Field for Primitive Data 12

Table 11: Values for the *measurement-quantity* Field for Structures 12

# Introduction

## Document Overview

This document specifies the naming standard for software identifiers and algorithm data and invocations that are used by T-engineering for embedded software. This document also specifies the data dictionary standard used to define the properties of algorithm data and invocations.

## Applicability

This standard is applicable to all embedded system applications with the exception of the libraries.

## Assumptions

1. …

# Naming Standard

A naming standard for C software identifiers and algorithm data and invocations improves the ability to read and maintain software, algorithm description documents and models by reducing the amount of variation in styles. A naming standard consists of fields, formats, and a formal syntax. This standard describes the set of fields and specifies the format for how these fields are combined to construct an object name.

## Naming Fields

A name is composed of a set of fields, each of which describes a specific characteristic. The following sections define the individual fields, provide instructions for determining their values, and indicate their applicability to classes of C software identifiers and algorithm data and invocations. The set of fields supported by this standard are as follows:

* ***SW-unit*** *[Defined by the name of the SW-unit defining the object]*
* ***storage-type*** *[Defined by the memory used]*
* ***data-type*** *[Defined by the data]*
* ***measurement-quantity*** *[Defined by representation of the object]*
* ***descriptive-abbreviation*** *[Defined by representation of the object]*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| *SW-unit* | *storage-type* | *data-type* | *underscore* | *measurement-quantity* | *underscore* | *descriptive-abbreviation* |
| **XXXX** | **x** | **x** | **\_** | **x** | **\_** | **XxxxXxx** |
| Maximum length is 31 | | | | | | |

Table 1: Naming Overview

### *SW-unit*

The *SW-unit* field specifies the ownership of an object, function, typedef name, data, or invocation. The *SW-unit* field shall equal to the SWU name.

### *storage-type*

The *storage-type* field specifies the type of memory used to store an object or data. The *storage-type* field shall be selected from the following tables.

|  |  |
| --- | --- |
| ***storage-type*** | **Description** |
| **c** | Constant   * Read-only data that is fixed at compile-time. * Within software, used for objects defined using the const type qualifier (that are not calibrations or slews), object-like macros, and enumeration constants. |
| **k** | Online Calibration   * Read-only data that may be emulated by instrumentation and is used to adjust the behavior of the software for different applications. |
| **m** | Offline Calibration (Secured)   * Read-only data that may be changed by instrumentation and is used to adjust the behavior of the software for different applications. * Read-only data which can be flashed but not emulated ore online calibrated, e.g. values representing safety relevant property while driving. [6] |
| **s** | System Constants   * System-defined constant. Read-only data that may be emulated by instrumentation and is used to adjust the behavior of the software for different applications. * Read-only data which can be downloaded and emulated but not calibrated, [6] |
| **i** | Instrumentation Slew   * Read-only data that may be emulated by instrumentation and is used to alter the behavior of the software during development. |
| **b** | Static (a.k.a. battery independent non-volatile static)   * Retains value with loss of ignition or battery (emulated EEPROM). * Retains value with controller reprogramming. * Integrity check performed on any reset. * Not updated after end of line programming (except via a service procedure). |
| **e** | Cumulative (a.k.a. battery independent non-volatile dynamic non-defaulted)   * Retains value with loss of ignition or battery (emulated EEPROM). * Retains value with controller reprogramming. * Integrity check performed on any reset. |
| **d** | Adaptive (a.k.a. battery independent non-volatile dynamic defaulted)   * Retains value with loss of ignition or battery (emulated EEPROM). * Defaulted with controller reprogramming. * Integrity check performed on any reset. |
| **r** | Preserved   * Retains value with loss of ignition (battery-backed RAM). * Defaulted with controller reprogramming. * Integrity check performed on any reset. |
| **n** | Expendable (a.k.a. ignition independent non-volatile)   * Retains value with loss of ignition (battery-backed RAM). * Defaulted with controller reprogramming. * Integrity check only performed on power-up reset following proper shutdown. |

Table 2: NVM Based Values for the *storage-type* Field

|  |  |
| --- | --- |
| ***storage-type*** | **Description** |
| **v** | Variable   * Read-write data. * Within software, used for objects with static storage duration (i.e., objects stored in fixed RAM locations). |
| **w** | Secure Variable   * Secure read-write data. * Within software, used for secure objects with static storage duration that must be stored separately from objects with *storage-type* of **V**. |
| **y** | Secure Dual Store Variable   * Secure read-write data maintained in lockable memory with a redundant copy. * Within software, used for secure objects with static storage duration that must be maintained as dual stores. |
| **l** | Local   * Temporary read-write data. * Within software, used for objects with automatic storage duration (i.e., objects stored in registers or on the stack). |
| **t** | Type   * Within algorithms, used for enumerated types. * Within software, used for typedef names and tags. |

Table 4: Type Based Values for the *storage-type* Field

### *data-type*

The *data-type* field provides information on the composition of an object or. The *data-type* field shall be selected from Table 5 below.

|  |  |
| --- | --- |
| ***data-type*** | **Description** |
| **e** | Element |
| **a** | Array   * Set of homogeneous data, accessed directly via an index. |
| **t** | Table   * Set of homogeneous data, accessed via interpolation. |
| **n** | Nonlinear Axis   * Set of homogeneous data, used to specify arbitrarily spaced breakpoints for a nonlinear axis of a table. |
| **s** | Structure   * Set of heterogeneous data. |
| **p** | Pointer   * Location of an object or function in memory. * Only applicable to software. |

Table 5: Values for the *data-type* Field

### *measurement-quantity*

The *measurement-quantity* field specifies the physical quantity associated with the item. The *measurement-quantity* field shall be selected from tables below.

| ***measurement-quantity*** | **ISO Measurement Quantity and Symbol** | **Units** |
| --- | --- | --- |
| **a** | Acceleration (*a*) | meters / second2 (m/s2) kilometers / hour / second (kph/s) |
| **A** | Area (*A*) | square meters (m2) square millimeters (mm2) |
| **c** | Specific heat (*c*) | kilojoules / kilogram Kelvin (kJ/kg K) |
| **C** | Capacitance (*C*) | farads (F) |
| **chrg** | Electric charge (*Q*) | ampere-hours (Ah) |
| **cp** | Specific molar heat at constant pressure (*cp*) | joules / mole Kelvin (J/mol K) |
| **cv** | Specific molar heat at constant volume (*cv*) | joules / mole Kelvin (J/mol K) |
| **E** | Energy (*E*) | joules (J) |
| **f** | Frequency (*f*) | hertz (Hz) |
| **F** | Force (*F*) | newtons (N) |
| **G** | Conductance (*G*) | siemens (S) |
| **H** | Enthalpy (*H*) | joules (J) |
| **I** | Electric current (*I*) | amperes (A) |
| **J** | Moment of inertia (*J*) | kilograms-square meters (kg m2) |
| **l** | Length (distance) (*l*) | meters (m) millimeters (mm) kilometers (km) |
| **L** | Inductance (*L*) | henrys (H) |
| **m** | Mass (*m*) | grams (g) milligrams (mg) kilograms (kg) |
| **M** | Moment of force (torque) (*M*) | newton-meters (Nm) |
| **mol** | Quantity (*n*) | moles (mol) |
| **molW** | Molecular weight (*M*) | grams / mole (g/mol) |
| **n** | Rotational frequency (speed) (*n*) | revolutions / minute (rpm) |
| **p** | Pressure (*p*) | kiloPascals (kPa) |
| **P** | Power (*P*) | watts (W)  kilowatts (kW) |
| **phi** | Angle (*φ*) | degrees (º) radians (rad) |
| **PHI** | Magnetic flux (*Φ*) | webers (Wb)  volts-seconds (V s) |
| **Q** | Heat (*Q*) | joules (J) |
| **R** | Electric resistance (*R*) | ohms (Ω) |
| **rho** | Density (*ρ*) | kilograms / cubic meter (kg/m3) |
| **S** | Entropy (*S*) | joules / Kelvin (J/K) |
| **t** | Time, duration, interval (period) (*t*) | seconds (s) milliseconds (ms) |
| **T** | Temperature (*t*) | degrees Celsius (ºC) |
| **TK** | Temperature (*T*) | Kelvin (K) |
| **U** | Electric potential (voltage) (*U*) | volts (V) millivolts (mV) |
| **v** | Velocity (speed) (*v*) | meters / second (m/s) kilometers / hour (kph) |
| **V** | Volume (*V*) | liters (l) milliliters (ml) |
| **w** | Angular velocity (speed) (*ω*) | radians / second (rad/s) |
| **W** | Work (*W*) | joules (J) |
| **X** | Reactance (*X*) | ohms (Ω) |
| **Z** | Impedance (*Z*) | ohms (Ω) |

Table 6: ISO Values for the *measurement-quantity* Field for Continuous Data

|  |  |  |
| --- | --- | --- |
| ***measurement-quantity*** | **Measurement Quantity** | **Units** |
| **Cf** | Conversion factor | *dependent upon the conversion* |
| **cmp** | Composite (a quantity with a composition of units that is not explicitly covered by another value for this field) | *dependent upon the composition of units* |
| **K** | Gain | *dependent upon the gain* |
| **Pct** | Percent | percent (%) |
| **ppm** | Parts per million | ppm |
| **r** | Ratio (a proportional relation of two quantities of the same unit) | none |
| **dm** | mass flow | grams / second (g/s) |
| **dM** | Derivative of torque | newton-meters / second (Nm/s) |
| **dmol** | Derivative of quantity | moles / second (mol/s) |
| **dn** | Derivative of rotational frequency | revolutions / minute / second (rpm/s) |
| **d2n** | Second derivative of rotational frequency | revolutions / minute / second2 (rpm/s2) |
| **dp** | Derivative of pressure | kiloPascals / second (kPa/s) |
| **dP** | Derivative of power | watts / second (W/s)  kilowatts / hour (kW/h) |
| **dPct** | Derivative of percent | percent / second (%/s) |
| **dphi** | Derivative of angle | degrees / second (º/s) |
| **d2phi** | Second derivative of angle | degrees / second2 (º/s2) |
| **dT** | Derivative of temperature | degrees Celsius / second (ºC/s) |
| **dV** | Derivative of volume | liters / second (l/s) |
| **dU** | Derivative of voltage | volts / second (V/s) |
| **dw** | Derivative of angular velocity | radians / second2 (rad/s2) |

Table 8: Values for the *measurement-quantity* Field for Derivatives of Continuous Data

|  |  |  |
| --- | --- | --- |
| ***measurement-quantity*** | **Measurement Quantity** | **Description** |
| **b** | Binary data | A discrete data element with two and only two possible values. |
| **e** | Enumerated data | A discrete data element with more than two possible values. |
| **i** | Index | A discrete data element used to reference an entry of an array. |
| **Cnt** | Count | counts |

Table 9: Values for the *measurement-quantity* Field for Discrete Data

|  |  |  |
| --- | --- | --- |
| ***measurement-quantity*** | **Measurement Quantity** | **Description** |
| **y** | Encoded data, 8-bit | byte |
| **d** | Encoded data, 16-bit | word |
| **g** | Encoded data, 32-bit | longword |

Table 10: Values for the *measurement-quantity* Field for Primitive Data

|  |  |  |
| --- | --- | --- |
| ***measurement-quantity*** | **Measurement Quantity** | **Description** |
| **h** | Heterogeneous composite data | Data that is composed of multiple elements of possibly different data types. |

Table 11: Values for the *measurement-quantity* Field for Structures

### *descriptive-abbreviation*

The descriptive-abbreviation field identifies the content of an object. The descriptive-abbreviation field shall be formed from the combination of words, abbreviations, acronyms, and numbers as described in the sections below.

#### Words

Words shall start with an uppercase letter, and all subsequent letters in a word shall be lowercase (e.g., “**Engine**” as opposed to “**ENGINE**” or “**engine**”). The transition from one word to another is designated by the sequence of a lowercase letter followed by an uppercase letter (e.g., “**EngineSpeed**” where “**Engine**” is the first word and “**Speed**” is the second word). An underscore can be used as a delimiter after an acronym (e.g. **EGR\_Temp**)

#### Abbreviations

As the letters allowed in the naming of an object is limited to 31, the words most often need to be abbreviated. There are no clear rules how to abbreviate a word but a best practice is to remove the words vowels, except if the first letter is a vowel, and only use the constants (following this recommendation **engine speed** would be abbreviated to **EngnSpd**, but **EngSpd** is also ok since it is still clear what the meaning is).

#### Acronyms

Acronyms shall only be used if the acronym is a well-known acronym in the industry. For example **EGR** (**E**xhaust **G**as **R**ecirculation) is allowed but it is not allowed to abbreviate engine speed to ES. Acronyms shall be written in uppercase letters only. If it is not in the end of the descriptive abbreviation it shall be followed by an underscore (e.g. **EGR\_Temp**)

#### Numbers

Numbers shall consist of digits (4=Ok, four=Not Ok). The letter “**n**” shall be used to represent a negative number (e.g., “**n40**” represents -40), and the letter “**p**” shall be used to represent a decimal point (e.g., “**14p7**” represents 14.7). A numerical range shall not be represented in the name**.**