Fixed-Point Toolbox Release Notes

Contents

Summary by Version

This table provides quick access to what's new in each version. For clarification, see "About Release Notes" on page 1, below.

Version (Release)	New Features and Changes	Version Compatibility Considerations	Fixed Bugs and Known Problems	Related Documentation at Web Site
Latest Version V2.0 (R2007a)	Yes Details	Yes Summary	Bug Reports Includes fixes	Printable Release Notes: PDF Current product documentation
V1.5 (R2006b)	Yes Details	No	Bug Reports Includes fixes	No
V1.4 (R2006a)	Yes Details	Yes Summary	Bug Reports Includes fixes	No
V1.3 (R14SP3)	Yes Details	Yes Summary	Bug Reports Includes fixes	No
V1.2 (R14SP2)	Yes Details	No	Bug Reports	No
V1.1 (R14SP1)	No	No	Yes Details	No
V1.0 (R14)	Yes Details	Not applicable	No bug fixes	No

About Release Notes

Use release notes when upgrading to a newer version to learn about new features and changes, and the potential impact on your existing files and practices. Release notes are also beneficial if you use or support multiple versions.

If you are not upgrading from the most recent previous version, review release notes for all interim versions, not just for the version you are installing. For example, when upgrading from V1.0 to V1.2, review the New Features and

Changes, Version Compatibility Considerations, and Bug Reports for V1.1 and V1.2.

New Features and Changes

These include

- New functionality
- Changes to existing functionality
- Changes to system requirements (complete system requirements for the current version are at the MathWorks Web site)
- Any version compatibility considerations associated with each new feature or change

Version Compatibility Considerations

When a new feature or change introduces a known incompatibility between versions, its description includes a **Compatibility Considerations** subsection that details the impact. For a list of all new features and changes that have compatibility impact, see the "Compatibility Summary for Fixed-Point Toolbox" on page 25.

Compatibility issues that become known after the product has been released are added to Bug Reports at the MathWorks Web site. Because bug fixes can sometimes result in incompatibilities, also review fixed bugs in Bug Reports for any compatibility impact.

Fixed Bugs and Known Problems

MathWorks Bug Reports is a user-searchable database of known problems, workarounds, and fixes. The MathWorks updates the Bug Reports database as new problems and resolutions become known, so check it as needed for the latest information.

Access Bug Reports at the MathWorks Web site using your MathWorks Account. If you are not logged in to your MathWorks Account when you link to Bug Reports, you are prompted to log in or create an account. You then can view bug fixes and known problems for R14SP2 and more recent releases.

The Bug Reports database was introduced for R14SP2 and does not include information for prior releases. You can access a list of bug fixes made in prior versions via the links in the summary table.

Related Documentation at Web Site

Printable Release Notes (PDF). You can print release notes from the PDF version, located at the MathWorks Web site. The PDF version does not support links to other documents or to the Web site, such as to Bug Reports. Use the browser-based version of release notes for access to all information.

Product Documentation. At the MathWorks Web site, you can access complete product documentation for the current version and some previous versions, as noted in the summary table.

Version 2.0 (R2007a) Fixed-Point Toolbox

This table summarizes what's new in Version 2.0 (R2007a):

New Features and Changes	Version Compatibility Considerations	Fixed Bugs and Known Problems	Related Documentation at Web Site
Yes Details below	Yes—Details labeled as **Compatibility Considerations**, below. See also Summary.	Bug Reports Includes fixes	Printable Release Notes: PDF Current product documentation

New features and changes introduced in this version are

Fast Execution for Fixed-Point Algorithms in MATLAB

The new Embedded MATLAB MEX functionality converts M-code to C-MEX functions. These C-MEX functions contain Embedded MATLAB optimizations for automatically accelerating fixed-point algorithms to compiled C code speed in MATLAB. For more information, refer to "Working with Embedded MATLAB MEX" in the Embedded MATLAB documentation.

New fi Syntaxes that Have fimath as an Argument

The following syntaxes have been added to the `fi` object:

- `a = fi(v,F)`
- `a = fi(v,s,F)`
- `a = fi(v,s,w,F)`
- `a = fi(v,s,w,f,F)`
- `a = fi(v,s,w,slope,bias,F)`
- `a = fi(v,s,w,slopeadjustmentfactor,fixedexponent,bias,F)`

where `v` is value, `s` is signedness, `w` is word length, `f` is fraction length, and `F` is a `fimath` object. Refer to "Working with fi Objects" or the `fi` reference page for more information.

Increased Support for Fixed-Point Toolbox Features in the Embedded MATLAB Subset

The following Fixed-Point Toolbox features are now supported by the Embedded MATLAB subset:

- Dot notation for getting the values of `fimath` properties
- `get` function for `fi` and `fimath` objects
- `diag`, `permute`, `tril`, and `triu` functions

For a complete list of the Fixed-Point Toolbox features supported by the Embedded MATLAB subset, refer to "Supported Functions and Limitations of Fixed-Point Embedded MATLAB".

Embedded MATLAB Enhanced to Support N-Dimensional Arrays and Function Handles

Embedded MATLAB now supports N-dimensional arrays and function handles.

`get` Function Must be Declared Extrinsic in Embedded MATLAB

There is a change to how you must use the `get` function in Embedded MATLAB to call properties of any object other than `fi` objects.

Compatability Consideration

To get properties of non-`fi` objects in Embedded MATLAB, you must first declare `get` to be an extrinsic function. As of this release, if you do not do so, your code will error. For more information, refer to "Calling MATLAB Functions" in the Embedded MATLAB documentation.

Embedded MATLAB Does Not Support & and | Operators

Embedded MATLAB no longer supports & and | operators in `if` and `while` conditional statements.

Compatablility Consideration

In prior releases, these operators compiled without error, but their short-circuiting behavior was not implemented correctly. Substitute && and || operators instead.

New Demo

The "Fixed-Point Lowpass Filtering Using Embedded MATLAB MEX" demo is new in this release. This demo steps you through generating a C-MEX function from M-code, running the generated C-MEX function, and displaying the results.

Version 1.5 (R2006b) Fixed-Point Toolbox

This table summarizes what's new in Version 1.5 (R2006b):

New Features and Changes	Version Compatibility	Fixed Bugs and Known Problems	Related Documentation at
Yes Details below	No	Bug Reports Includes fixes	No

New features and changes introduced in this version are

Licensing Changes

You now can use `fi` objects with the `DataType` property set to `double` *without* a Fixed-Point Toolbox license when the `fipref` `LoggingMode` property is set to `off`. For details about the Fixed-Point Toolbox licensing model, refer to "Licensing" in the product documentation.

Fixed-Point Square Root Support

In this release, fixed-point square root support has been added to

- Fixed-Point Toolbox, via the `sqrt` function
- Embedded MATLAB, via support for the Fixed-Point Toolbox `sqrt` function
- Simulink, via fixed-point support for the `sqrt` mode of the Math Function block

These products use the same bisection algorithm to implement their fixed-point square root functionality and yield identical results.

Limited Dot Notation Support Added to Fixed-Point Embedded MATLAB

Dot notation is now supported in Embedded MATLAB for getting the values of `numerictype` object properties. Dot notation is not supported for `fi` or `fimath` objects, and it is not supported for setting properties.

get Function Support Added to Fixed-Point Embedded MATLAB

The Fixed-Point Toolbox `get` function is now supported for use with Embedded MATLAB with the following limitations:

- Only supported for use with `numerictype` objects
- The syntax `structure = get(o)` is not supported

New Default Syntax for fi Object

You can now use the syntax `fi` without any input arguments to return a default `fi` object with no value, 16-bit word length, and 15-bit fraction length.

Version 1.4 (R2006a) Fixed-Point Toolbox

This table summarizes what's new in Version 1.4 (R2006a):

New Features and Changes	Version Compatibility Considerations	Fixed Bugs and Known Problems	Related Documentation at Web Site
Yes Details below	Yes—Details labeled as **Compatibility Considerations**, below. See also Summary.	Bug Reports Includes fixes	No

New features and changes introduced in this version are

[Slope Bias] Math Support Added

Arithmetic using the `+`,`-`, `.*`, and `*` operators is now supported for objects with [Slope Bias] scaling. Refer to "[Slope Bias] Arithmetic" in the product documentation for more information.

In support of this feature, the following properties have been added to the `fimath` object:

- `ProductBias` — Bias of the product data type
- `ProductFixedExponent` — Fixed exponent of the product data type
- `ProductSlope` — Slope of the product data type
- `ProductSlopeAdjustmentFactor` — Slope adjustment factor of the product data type
- `SumBias` — Bias of the sum data type
- `SumFixedExponent` — Fixed exponent of the sum data type
- `SumSlope` — Slope of the sum data type
- `SumSlopeAdjustmentFactor` — Slope adjustment factor of the sum data type

Refer to "Property Reference" in the product reference documentation for more information.

Scaled Double Data Type Support Added to the fi Object

The `fi` object now supports the scaled double data type. The value `ScaledDouble` has been added to the `DataType` property of the `numerictype` object. The following values have also been added to the `DataTypeMode` property of the `numerictype` object:

- `Scaled double: binary point scaling`
- `Scaled double: slope and bias scaling`
- `Scaled double: unspecified scaling`

Math operations are supported for `fi` objects with data type `ScaledDouble`.

Global DataTypeOverride Property Added to the fipref Object

The `fipref` object now has the property `DataTypeOverride`, which allows you to override `fi` objects with scaled doubles, singles, or doubles. Refer to "Using fipref Objects to Set Data Type Override Preferences" in the product documentation for more information.

Embedded MATLAB Supports More Fixed-Point Toolbox Functions

The following Fixed-Point Toolbox functions are now supported by Embedded MATLAB:

- `bitand`
- `bitcmp`
- `bitget`
- `bitor`
- `bitset`
- `bitshift`
- `bitxor`
- `rescale`

Refer to "Supported Functions and Limitations of Fixed-Point Embedded MATLAB" in the product documentation for more information.

Embedded MATLAB Does Not Support a CastBeforeSum Value of `'false'`

You can no longer set the `fimath` object property `CastBeforeSum` to `false` or `0` in Embedded MATLAB Function blocks. The reason for this restriction is that Embedded MATLAB does not produce the same numerical results as MATLAB when `CastBeforeSum` is `false`.

Compatibility Considerations

In the previous release, `CastBeforeSum` was set to `false` for default `fimath` objects in Embedded MATLAB. If you have existing models that contain Embedded MATLAB Function blocks in which `CastBeforeSum` is `false`, you will now get an error when you compile or update your model. To correct this issue, you must set `CastBeforeSum` to `true`. To automate this process, you can run the utility `slupdate` either from the Model Advisor or by typing the following command at the MATLAB command line:

```
slupdate ('modelname')
```

where '*modelname*' is the name of the model containing the Embedded MATLAB Function block that generates the error. `slupdate` prompts you to update this property by selecting one of these options:

Option	Action
Yes	Updates the first occurrence of `CastBeforeSum=false` in Embedded MATLAB Function blocks in the model and then prompts you for each subsequent instance found in the model.
No	Does not update any occurrences of `CastBeforeSum=false` in the model.
All	Updates all occurrences of `CastBeforeSum=false` in the model.

Note `slupdate` detects `CastBeforeSum=false` only in *default* `fimath` objects defined for Simulink signals in Embedded MATLAB Function blocks. If you modified the `fimath` object in an Embedded MATLAB Function block, update `CastBeforeSum` manually in your model and fix the errors as they are reported.

`'round'` Value Added to the fimath Object RoundMode Property

The `RoundMode` property value `round` has been added to the `fimath` object. The behavior of this rounding mode is identical to the MATLAB `round` function. For more information refer to "RoundMode" in the product documentation.

numerictype Object Syntax Change

Previously, if you created a `numerictype` object without specifying a value for the `FractionLength` property, the fraction length would be automatically set to `15`. Now however, if you do not set the `FractionLength` property when creating a `numerictype` object, the scaling will remain unspecified. For example:

```
T = numerictype(1, 16)

T =
```

```
          DataTypeMode: Fixed-point: unspecified scaling
                Signed: true
            WordLength: 16

T.scaling

ans =

Unspecified

T.FractionLength

ans =

     0
```

Compatibility Considerations

Any instances of this syntax in your existing code will now return a different result.

Minimums and Maximums Now Logged After Quantization

Previously, the `fi` and `quantizer` objects logged minimums and maximums before quantization. They now log after quantization.

Compatibility Considerations

If your fixed-point data overflows and you want to log minimums and maximums for the full floating-point range, use the `'ScaledDoubles'` or `'TrueDoubles'` values of the `fipref` object `DataTypeOverride` property. For example, the following fixed-point variable overflows. The saturated minimum and maximum values are logged:

```
p = fipref;
p.LoggingMode = 'On';
p.DataTypeOverride = 'ForceOff';
```

```
a = fi(-2:2, true, 16, 15)
Warning: 3 overflows occurred in the fi assignment operation.

a =

          -1           -1            0      0.99997      0.99997

          DataTypeMode: Fixed-point: binary point scaling
                Signed: true
            WordLength: 16
        FractionLength: 15

             RoundMode: nearest
          OverflowMode: saturate
           ProductMode: FullPrecision
  MaxProductWordLength: 128
               SumMode: FullPrecision
      MaxSumWordLength: 128
         CastBeforeSum: true

logreport(a)

      minlog      maxlog   lowerbound   upperbound   noverflows   nunderflows
  a       -1   0.9999695           -1    0.9999695            3             0
```

Now set `DataTypeOverride` to `'ScaledDoubles'`. Note that overflows are reported, but the data is not quantized. The minimum and maximum logs show the full possible range of the data without quantization:

```
p = fipref;
p.LoggingMode = 'On';
p.DataTypeOverride = 'ScaledDoubles';

b = fi(-2:2, true, 16, 15)
Warning: 3 overflows occurred in the fi assignment operation.

b =

    -2    -1     0     1     2
```

```
          DataTypeMode: Scaled double: binary point scaling
                Signed: true
            WordLength: 16
        FractionLength: 15

             RoundMode: nearest
          OverflowMode: saturate
           ProductMode: FullPrecision
  MaxProductWordLength: 128
               SumMode: FullPrecision
      MaxSumWordLength: 128
         CastBeforeSum: true

logreport(b)

      minlog   maxlog   lowerbound   upperbound   noverflows   nunderflows
  b       -2        2           -1    0.9999695            3             0
```

For an in-depth example of using logging and data type override to help set appropriate scalings for fixed-point quantities, see the Fixed-Point Toolbox "Fixed-Point Data Type Override, Min/Max Logging, and Scaling" demo.

Version 1.3 (R14SP3) Fixed-Point Toolbox

This table summarizes what's new in Version 1.3 (R14SP3):

New Features and Changes	Version Compatibility Considerations	Fixed Bugs and Known Problems	Related Documentation at Web Site
Yes Details below	Yes—Details labeled as **Compatibility Considerations**, below. See also Summary.	Bug Reports Includes fixes	No

New features and changes introduced in this version are

Fixed-Point Toolbox Function Support Added to Embedded MATLAB

The Embedded MATLAB Function block lets you compose a MATLAB language function in a Simulink model that generates embeddable code using the Embedded MATLAB subset. When you simulate the model or generate code for a target environment, a function in an Embedded MATLAB Function block generates efficient C code. This code meets the strict memory and data type requirements of embedded target environments. In this way, Embedded MATLAB Function blocks bring the power of MATLAB for the embedded environment into Simulink.

For more information about the Embedded MATLAB Function block and the Embedded MATLAB subset, refer to the following documentation:

- Embedded MATLAB Function block reference page in the Simulink documentation
- "Using the Embedded MATLAB Function Block" in the Simulink documentation
- "Working with Embedded MATLAB" in the Embedded MATLAB documentation

You can now use a significant number of Fixed-Point Toolbox functions with Embedded MATLAB. Refer to "Supported Functions and Limitations of Fixed-Point Embedded MATLAB" in the Using Fixed-Point Toolbox documentation.

Note To simulate models using fixed-point data types in Simulink, including when using the Embedded MATLAB Function block, you must have a Simulink Fixed Point license.

Double, Single, and Boolean Data Type Support Added to the fi Object

The `fi` object now supports `double`, `single`, and `boolean` data types. The values `double`, `single`, and `boolean` have been added to the `DataType` and `DataTypeMode` properties of the `numerictype` object. Math operations are supported for `fi` objects with data type `single` or `double`, but not `boolean`.

Fixed-Point Doubles Override, Min/Max Logging, and Scaling Demo

Since floating-point data types are now supported in Fixed-Point Toolbox, it is possible to use doubles override and min/max scaling to help you choose the appropriate scalings for fixed-point variables in your algorithms. This is especially helpful when converting a floating-point algorithm to fixed point. A new demo "Fixed-Point Doubles Override, Min/Max Logging, and Scaling" leads you through an example of this process. You can access this demo from the **Demos** pane of the Help browser under Toolboxes > Fixed-Point.

Helper Functions Added for Accessing Logged Information

In the previous release it became possible to log overflows and underflows as warnings for all assignment, plus, minus, and multiplication operations when the `fipref` `LoggingMode` property is set to `on`. Now when `LoggingMode` is `on`, you can also use the following helper functions to return logged information to you at the MATLAB command line:

- `maxlog` — Returns the maximum real-world value
- `minlog` — Returns the minimum real-world value
- `noperations` — Returns the number of quantized operations
- `noverflows` — Returns the number of overflows
- `nunderflows` — Returns the number of underflows

To clear the log, use the function `resetlog`.

RoundMode Property Value 'round' Now Called 'nearest'

The `RoundMode` property value `round` is now `nearest`. This is a reflection of the fact that this rounding mode is identical to the Simulink rounding mode `round toward nearest`, and different from the behavior of the MATLAB `round` function.

Compatibility Considerations

For this release, any code using the `RoundMode` property value `round` will still work as it did in previous releases. However, you should update each instance of the property value `round` to `nearest` because in a later release, the property value `round` will give a different answer.

Version 1.2 (R14SP2) Fixed-Point Toolbox

This table summarizes what's new in Version 1.2 (R14SP2):

New Features and Changes	Version Compatibility Considerations	Fixed Bugs and Known Problems	Related Documentation at Web Site
Yes Details below	No	Bug Reports	No

New features and changes introduced in this version are

Overflow and Underflow Logging

Fixed-Point Toolbox now allows you to log overflows and underflows as warnings for all assignment, plus, minus, and multiplication operations. Refer to "Using fipref Objects to Set Logging Preferences" in the Fixed-Point Toolbox documentation for more information.

New Functions

The following functions are new in Fixed-Point Toolbox 1.2:

abs	all	and	any	area
bar	barh	buffer	clabel	comet
comet3	compass	coneplot	contour	contour3
contourc	contourf	diag	end	errorbar
etreeplot	ezcontour	ezcontourf	ezmesh	ezplot
ezplot3	ezpolar	ezsurf	ezsurfc	feather
fplot	gplot	hankel	hist	histc
intmin	ipermute	isnumeric	isobject	line
logical	lowerbound	mesh	meshc	meshz

not	numberofelements	or	patch	pcolor
permute	plot3	plotmatrix	plotyy	polar
pow2	quiver	quiver3	rgbplot	ribbon
rose	scatter	scatter3	sdec	sign
slice	spy	stairs	stem	stem3
streamribbon	streamslice	streamtube	sum	surf
surfc	surfl	surfnorm	text	toeplitz
treeplot	tril	trimesh	triplot	trisurf
triu	uplus	upperbound	voronoi	voronoin
waterfall	xlim	ylim	zlim	

Version 1.1 (R14SP1) Fixed-Point Toolbox

This table summarizes what's new in Version 1.1 (R14SP1):

New Features and Changes	Version Compatibility Considerations	Fixed Bugs and Known Problems	Related Documentation at Web Site
No	No	Yes Details below	No

The particularly important bug fixes in this version are

Bitwise Operators Return Correct Answers for [Slope Bias] Signals

In the previous release, bitwise functions such as `bitshift` might have given wrong answers for [Slope Bias] fixed-point signals. This has been corrected in this release.

fi Object Operations with an Empty Array Work Properly

In the previous release, a segmentation violation occurred for any operation with the format

```
a op e
```

where `a` is a `fi` object, `e` is an empty array, and `op` is any operator such as `+`, `-`, `*`, `.*`, `<`, `>`, etc. This has been corrected in this release.

ispropequal Returns Correct Answers for fimath Objects

The `ispropequal` function has been updated to work properly in this release.

Version 1.0 (R14) Fixed-Point Toolbox

This table summarizes what's new in Version 1.0 (R14):

New Features and Changes	Version Compatibility Considerations	Fixed Bugs and Known Problems	Related Documentation at Web Site
Yes Details below	No	No bug fixes	No

Fixed-Point Toolbox provides fixed-point data types in MATLAB and enables algorithm development by providing fixed-point arithmetic. Fixed-Point Toolbox enables you to create the following types of objects:

- `fi` — Defines a fixed-point numeric object in the MATLAB workspace. Each `fi` object is composed of value data, a `fimath` object, and a `numerictype` object
- `fimath` — Governs how overloaded arithmetic operators work with `fi` objects
- `fipref` — Defines the display attributes for `fi` objects
- `numerictype` — Defines the data type and scaling attributes of `fi` objects
- quantizer — Quantizes data sets

Features

Fixed-Point Toolbox provides you with

- The ability to define fixed-point data types, scaling, and rounding and overflow methods in the MATLAB workspace
- Bit-true real and complex simulation
- Basic fixed-point arithmetic with binary point-only signals
  - Arithmetic operators `+`, `-`, `*`, `.*`
  - Division using the `divide` function
- Arbitrary word length up to `intmax('uint16')`

- Relational, logical, and bitwise operators
- Data visualization via the `plot` function
- Statistics functions such as `abs`, `max`, and `min`
- Conversions between binary, hex, double, and built-in integers
- Interoperability with Simulink, Signal Processing Blockset, and Filter Design Toolbox
- Compatibility with the Simulink To Workspace and From Workspace blocks

Getting Help

This section tells you how to get help for Fixed-Point Toolbox in this document and at the MATLAB command line.

Getting Help in the Fixed-Point Toolbox User's Guide

The objects of Fixed-Point Toolbox are discussed in the following chapters:

- "Working with fi Objects"
- "Working with fimath Objects"
- "Working with fipref Objects"
- "Working with numerictype Objects"
- "Working with quantizer Objects"

To get in-depth information about the properties of these objects, refer to "Property Reference".

To get in-depth information about the functions of these objects, refer to Function Reference.

Getting Help at the MATLAB Command Line

To get command-line help for Fixed-Point Toolbox objects, type

```
help objectname
```

For example:

```
help fi
help fimath
help fipref
help numerictype
help quantizer
```

To invoke Help Browser documentation for Fixed-Point Toolbox functions from the MATLAB command line, type

```
doc fixedpoint/functionname
```

For example:

```
doc fixedpoint/int
doc fixedpoint/add
doc fixedpoint/savefipref
doc fixedpoint/quantize
```

Compatibility Summary for Fixed-Point Toolbox

This table summarizes new features and changes that might cause incompatibilities when you upgrade from an earlier version, or when you use files on multiple versions. Details are provided in the description of the new feature or change.

Version (Release)	New Features and Changes with Version Compatibility Impact
Latest Version V2.0 (R2007a)	See the **Compatibility Considerations** subheading for each of these new features or changes: • "`get` Function Must be Declared Extrinsic in Embedded MATLAB" on page 6 • "Embedded MATLAB Does Not Support & and | Operators" on page 6
V1.5 (R2006b)	None
V1.4 (R2006a)	See the **Compatibility Considerations** subheading for each of these new features or changes: • "Embedded MATLAB Does Not Support a CastBeforeSum Value of `'false'`" on page 11 • "numerictype Object Syntax Change" on page 12 • "Minimums and Maximums Now Logged After Quantization" on page 13

Version (Release)	New Features and Changes with Version Compatibility Impact
V1.3 (R14SP3)	See the **Compatibility Considerations** subheading for this new feature or change: • "RoundMode Property Value 'round' Now Called 'nearest'" on page 18
V1.2 (R14SP2)	None
V1.1 (R14SP1)	None
V1.0 (R14)	Not applicable