

# Latest Features in Simulink Fixed Point

March 2009

**R2009a**

# Discrete Filter Support for Fixed Point

## Challenge

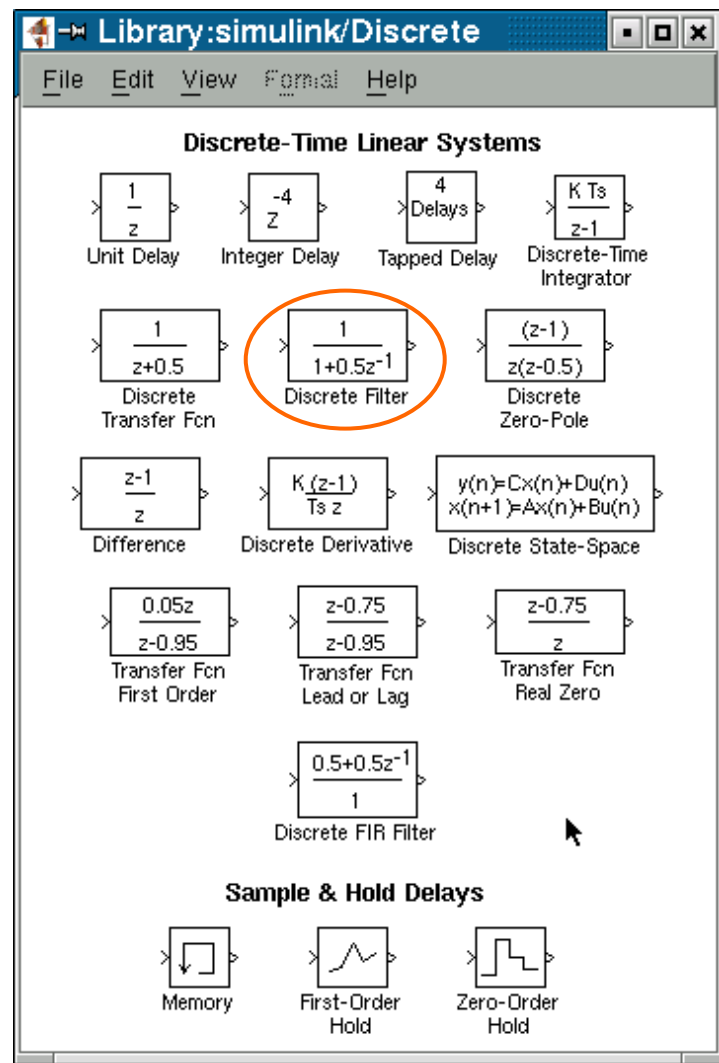
- The Discrete Filter block supported only floating-point, scalar, and real inputs.

## Solution

- Add support for signed fixed-point and integer data types
- Add support for vector and 2-D inputs
- Allow arbitrary input and coefficient complexity

## Benefit

- Data type, dimensions, and complexity support have been enhanced.
- Discrete Filter block can support a subset of usages for the Discrete Transfer Function, which does not support fixed point.



# Prelookup Block Breakpoint Data

## Challenge

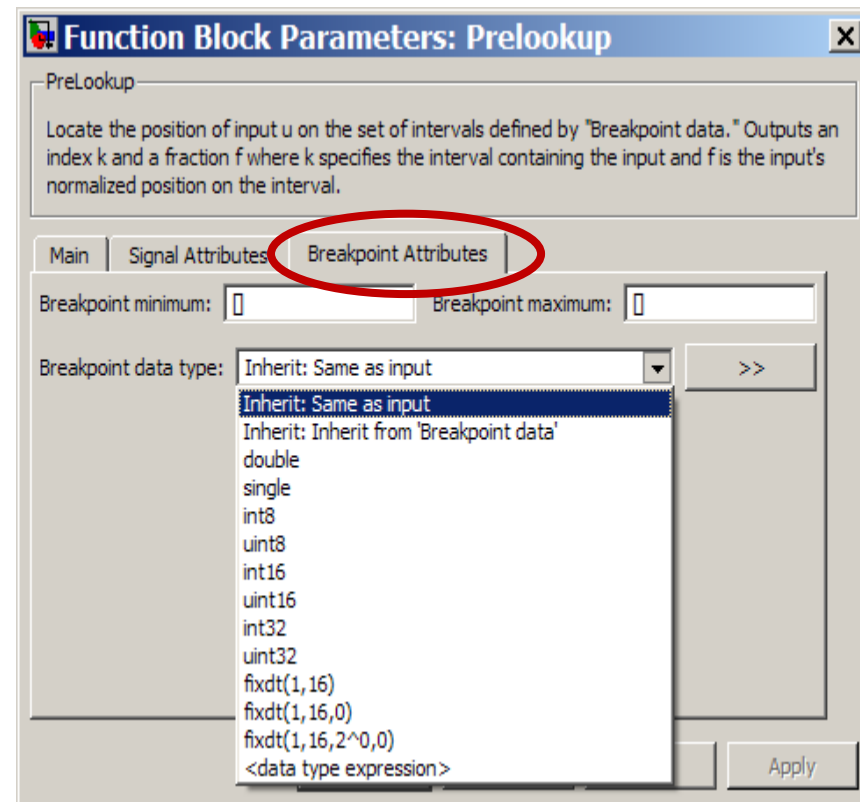
- Breakpoint data was converted to input data type, which could result in non-monotonic breakpoint data (causing compile error).
- Breakpoint data could not be shared between blocks with different input data types.

## Solution

- Separate breakpoint data type from input data type

## Benefit

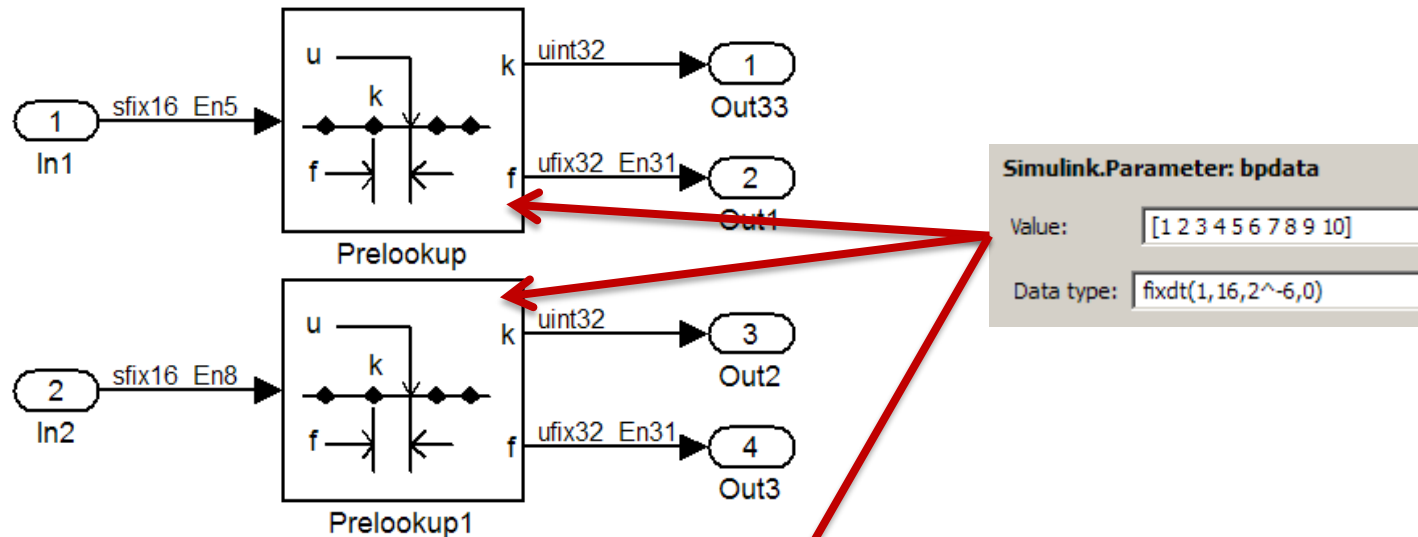
- Improves usability
- Reduces memory (RAM) usage by sharing data between blocks in the generated code
- Facilitates legacy code integration



# Prelookup Block Breakpoint Data

## Example

Breakpoint data shared between two Prelookup blocks with different input data types



```

16  /* Exported block parameters */
17  int16_T bpdata[10] = { 64, 128, 192, 256, 320, 384, 448, 512, 576, 640 } ; /* Variable: bpdata
18  * Referenced by blocks:
19  * '<Root>/Prelookup'
20  * '<Root>/Prelookup1'
21  */
    
```

# Fixed-Point Tool Support for Simulink Signal Objects

## Challenge

- Autoscaling tools did not support Simulink signal objects.

## Solution

- Enhance Fixed-Point Tool and Fixed-Point Advisor to support Simulink signal objects
- Provide data type analysis and reporting via Autoscale dialog

## Benefit

- Users can incorporate Simulink signal objects into their autoscaling workflow to maintain a data type–agnostic model.
- Data dictionary–driven workflows are improved.

Contents of: sigC2 (mmo)

Name	ProposedDT	Acc	DesignMin	DesignMax	ProposedMin	ProposedMax	Sim
c1 (base)	fixdt(1,16,12)	<input checked="" type="checkbox"/>	-2	4	-8	7.9998	
c2 (other)	fixdt(1,16,12)	<input checked="" type="checkbox"/>	-8	4	-8	7.9998	

Autoscale Information

**c1 (base)**

**Proposed Data Type Summary:**  
 This is a Simulink.Signal object defined in the base workspace.  
 Autoscaling proposed moving 9 bits from the range end to the precision end.  
 - Range will decrease and be 512 times smaller.  
 - Precision will increase and be 512 times finer.

**Shared Data Type Summary:**  
 There is a requirement for the data type of this result to match the data type of other results.  
[Highlight Blocks Sharing Same Data Type](#)  
[Highlight Blocks Connected to the Signal Object](#)

	Value	Percent Proposed Representable
Currently Specified Data	fixdt(1,16,3)	

# New Rounding Mode in Simulink Blocks

## Challenge

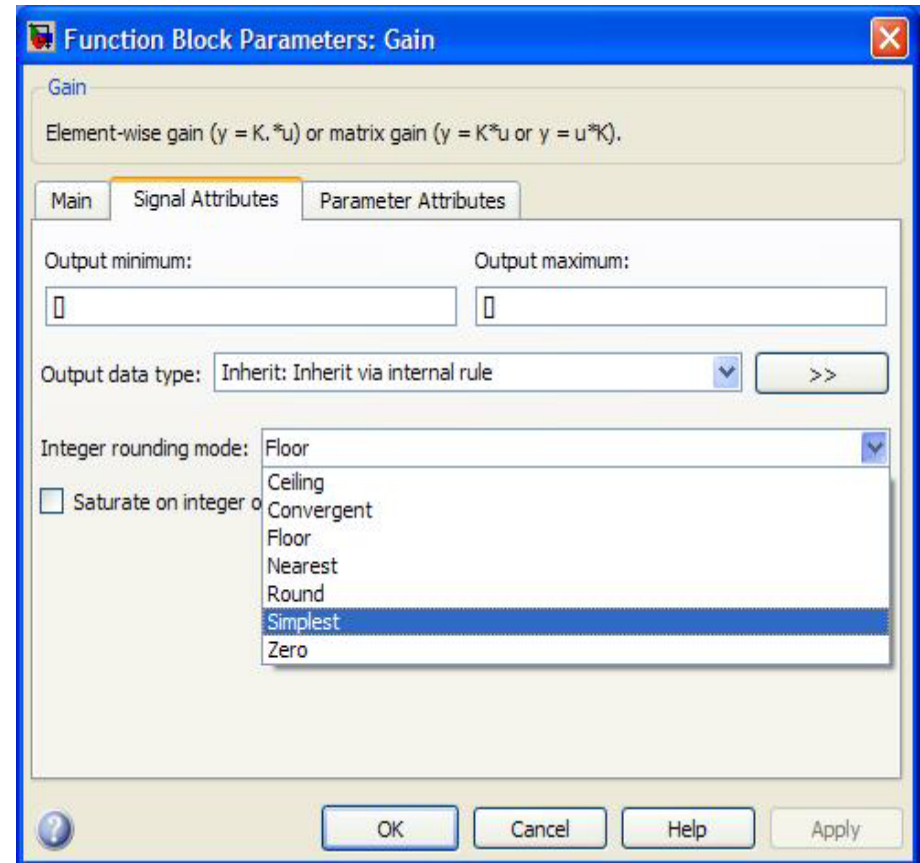
- Not every block supported the most efficient rounding method.

## Solution

- Add “Simplest” rounding mode to Simulink blocks that provide an “Integer rounding mode” parameter

## Benefit

- Gives users more flexibility in the tradeoff between cost and performance in their design
- Supports simulation and code generation



# Multiword Generated Code Improvements

## Challenge

- Multiword code was based on static C functions and required excessive temporary variables.

## Solution

- Dynamically generate multiword code in same file as algorithm code and leverage code generation optimization

## Benefit

- Code is more efficient.
- Code is easier to review.
- Compile times are faster.

```

:
MultiWordAdd(&a.chunks[0U], &b.chunks[0U], &rtb_Sum.chunks[0U], 4);
sLong2MultiWord(rtU.In3, &b.chunks[0U], 4);
MultiWordAdd(&rtb_Sum.chunks[0U], &b.chunks[0U], &a.chunks[0U], 4);
MultiWordSub(&rtb_Sum.chunks[0U], &a.chunks[0U], &b.chunks[0U], 4);
:
  
```