

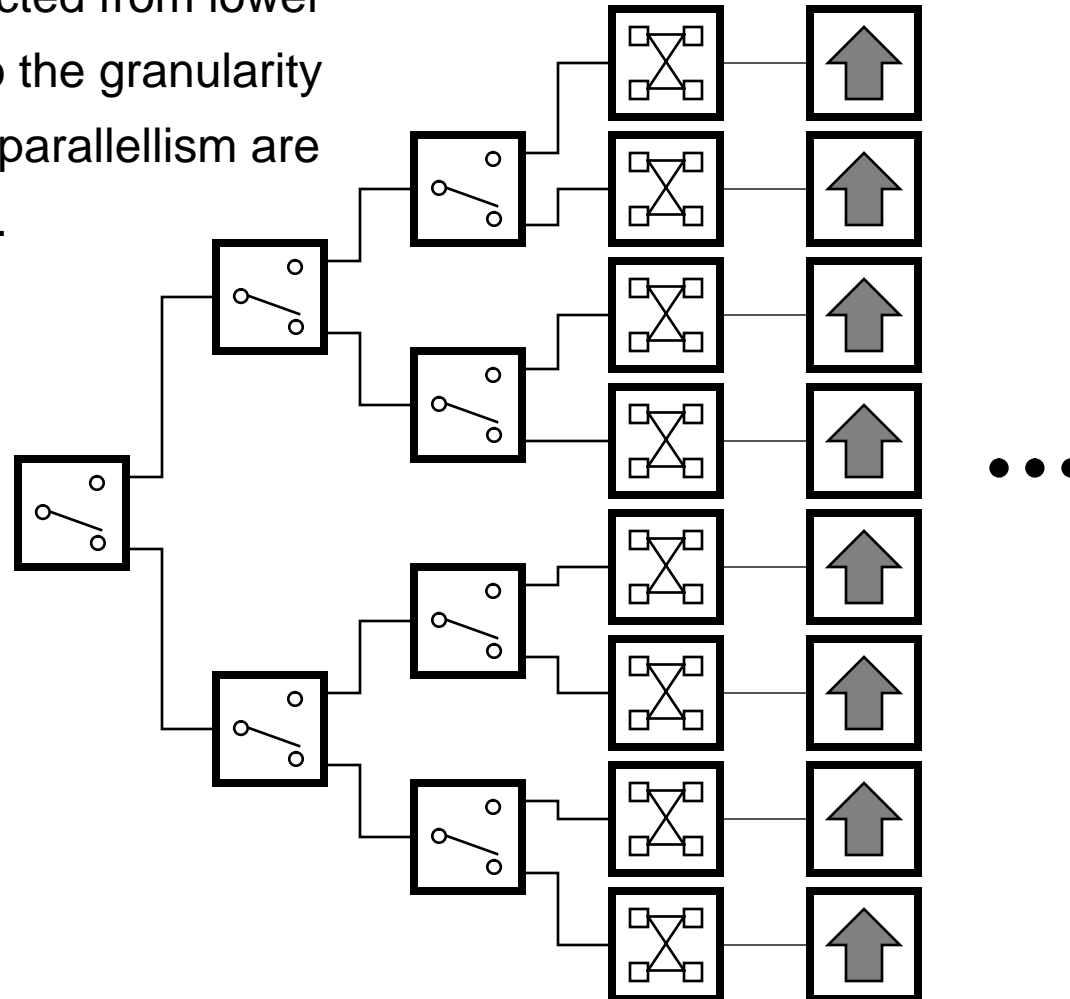
## Conclusions

### **Advantages of higher-order functions:**

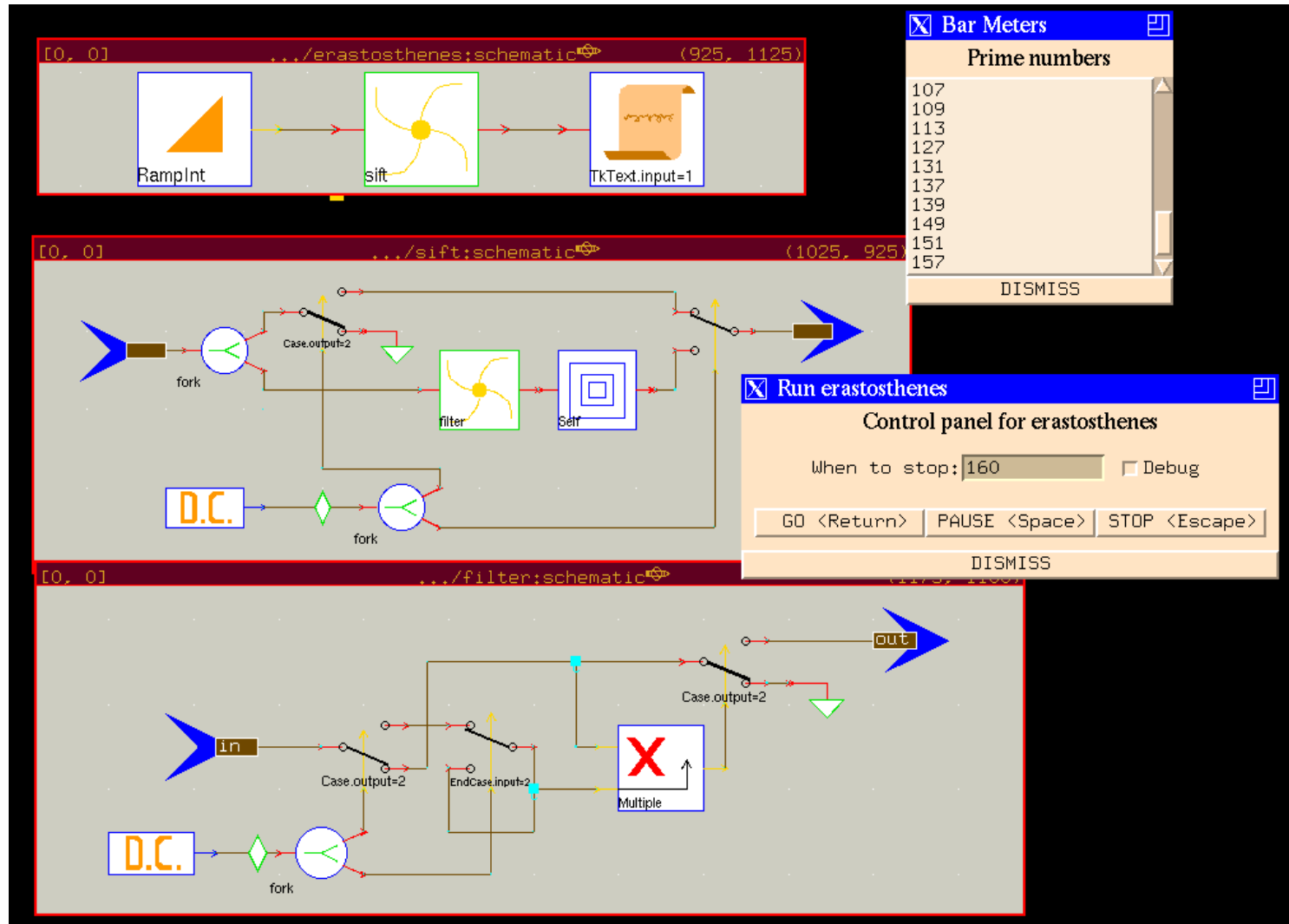
- **Parameterizable parallelism**
- **Improved program modularity**
- **Scalable programming**
- **Recursion**
- **Zero run-time cost in most cases**
- **Expressiveness of a visual syntax**

## In SDF, recursion is statically unrolled

In our example, a high-order FFT is constructed from lower order FFTs. So the granularity and degree of parallelism are parameterized.

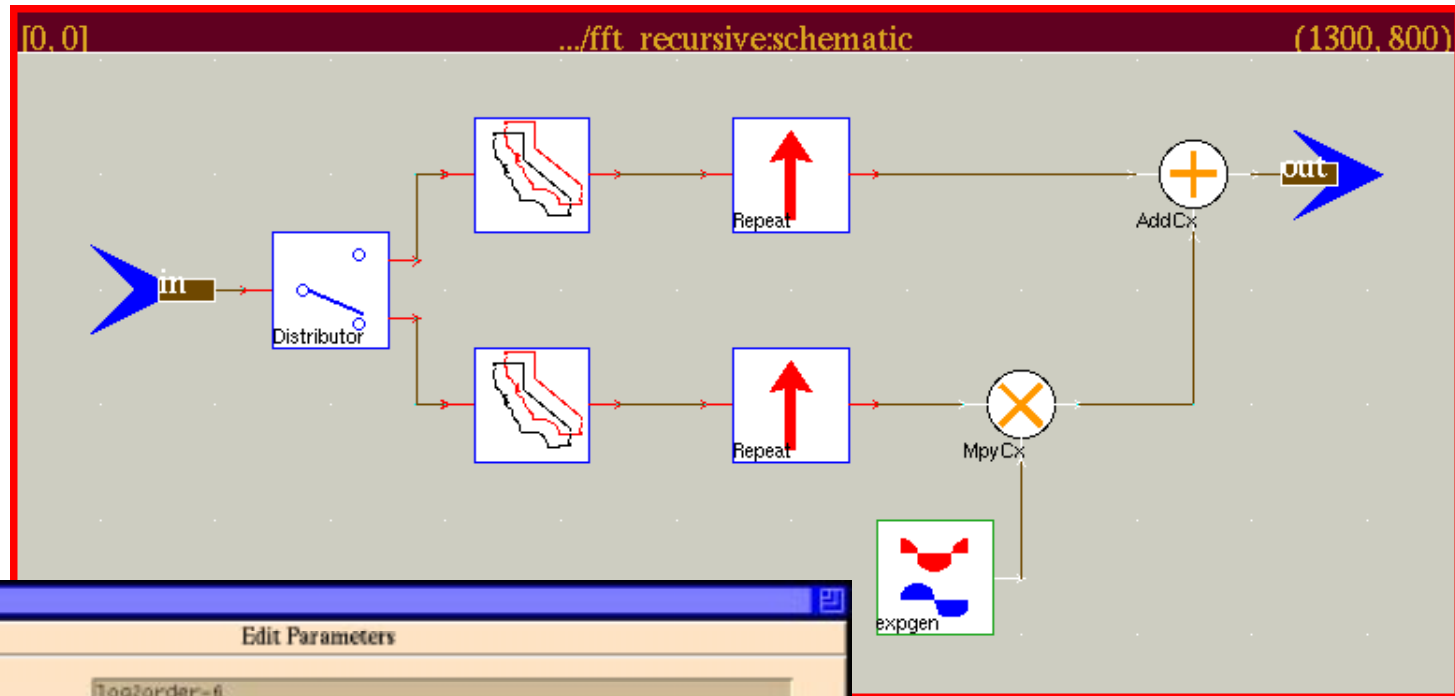


# Using Map for Recursion in Dynamic Dataflow



The “sieve of Erastosthenes” is a **highly irregular recursive algorithm** for computing prime numbers. The DDF domain in Ptolemy can model algorithms of this type.

# Using IfThenElse for Recursion



**Edit Params**

Edit Parameters

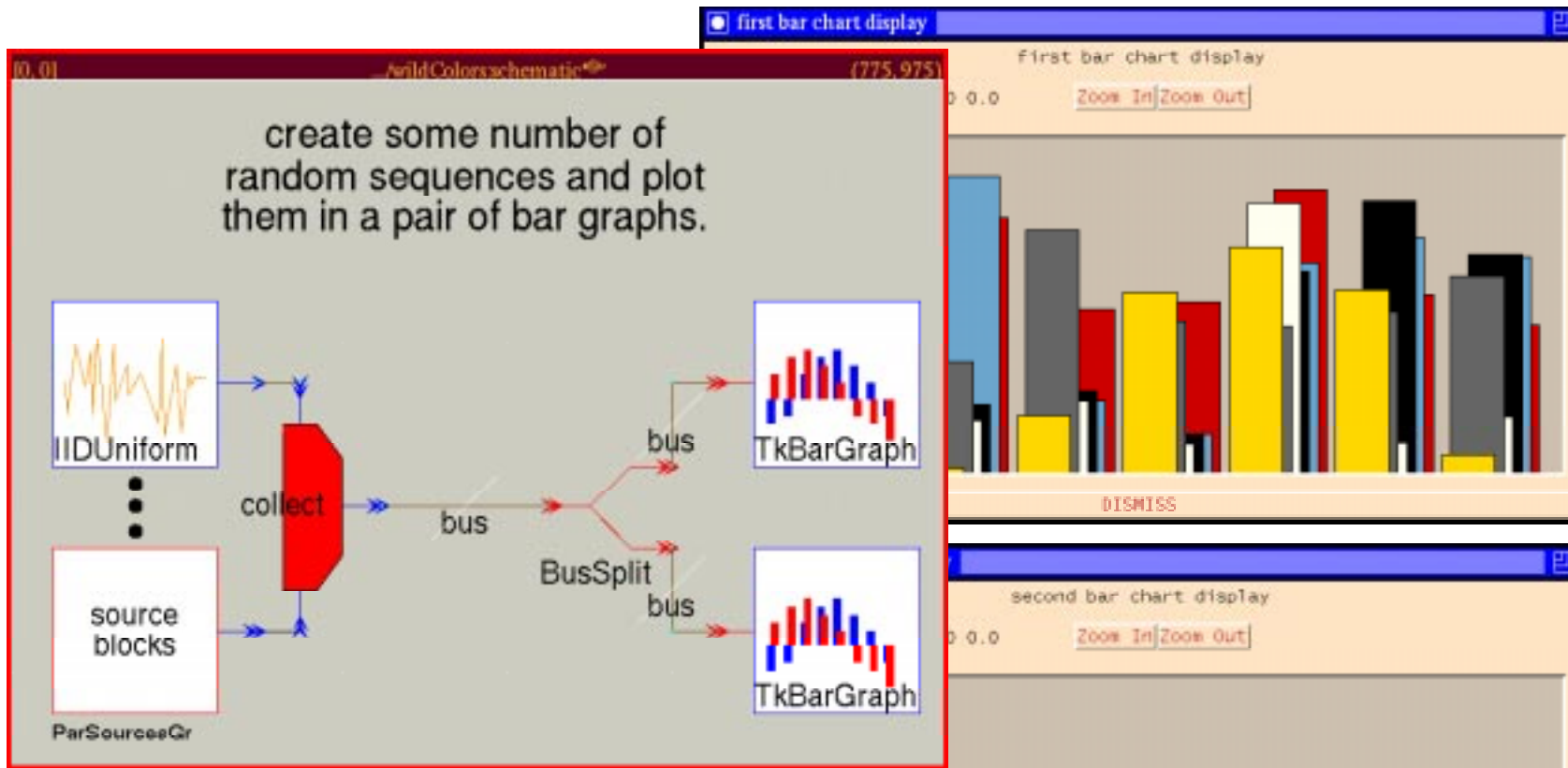
condition:	log2order=6
true_block:	fft_recursive
where_true_defined:	
true_input_map:	in
true_output_map:	out
true_parameter_map:	log2order = log2order-1
false_block:	FFTCk
where_false_defined:	
false_input_map:	input
false_output_map:	output
false_parameter_map:	order = 5 size = 32

OK Apply Close Cancel

## Recursive FFT

granularity of the bottom-level operation is arbitrary  
recursion can be unraveled at compile time, avoiding any run-time overhead

# Generalization of Map to Signal Sources

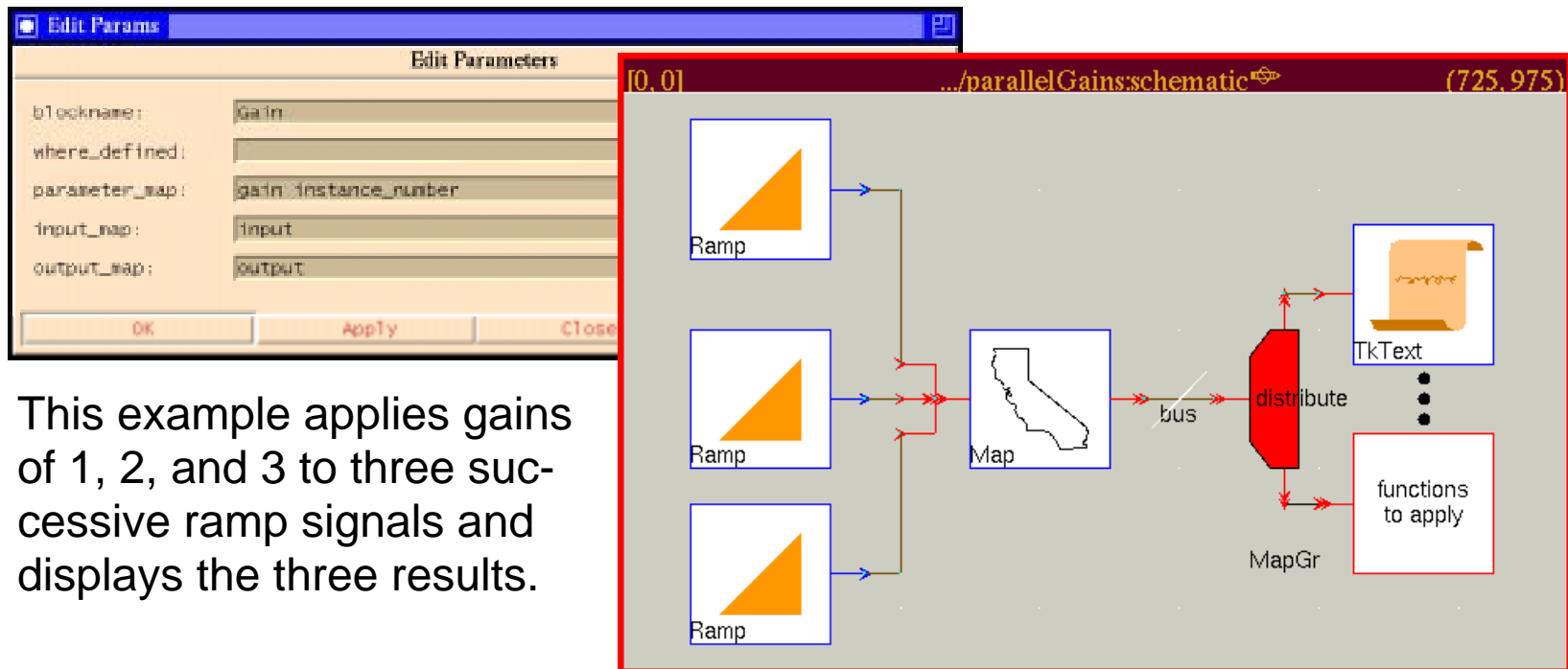


The ParSourcesGr (now called SrcGr) is a Map star with no inputs. The number of instances is determined by the number of outputs. The BusSplit is also an HOF star.

## Higher-Order Functions in Ptolemy

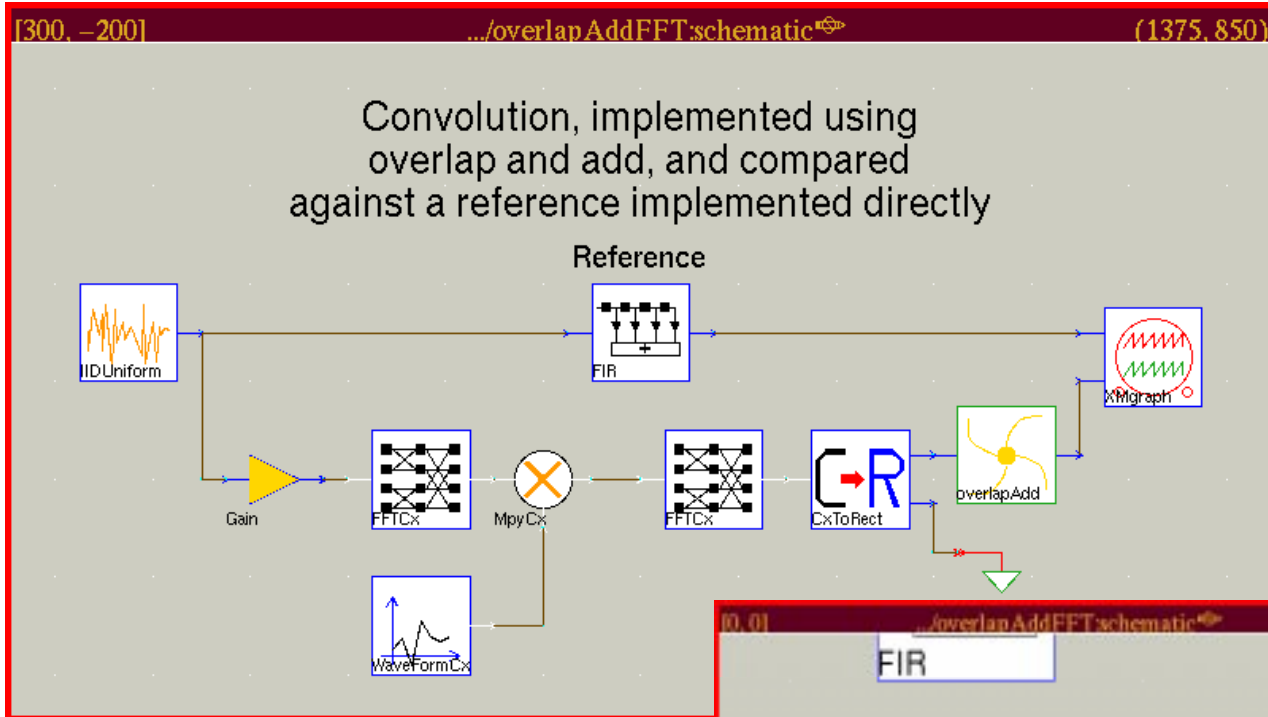
Examples implemented in Ptolemy:

- **Map** (apply a specified function to all inputs)
- **MapGr** (graphical version of Map)
- **IfThenElse** (apply one of two specified functions)
- **Chain** (apply a specified function sequentially)

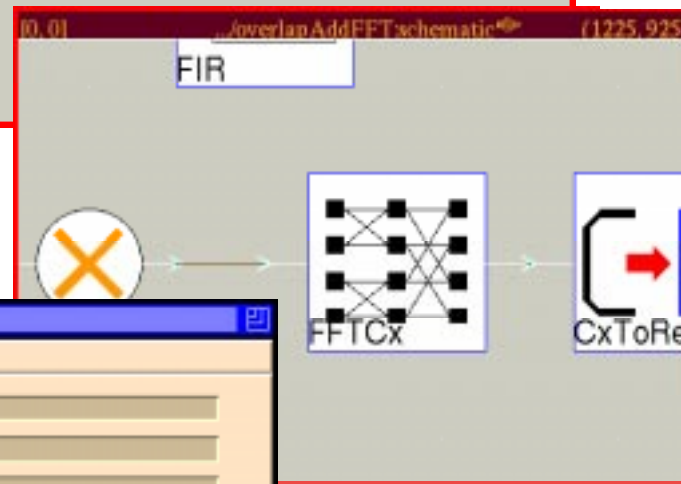


This example applies gains of 1, 2, and 3 to three successive ramp signals and displays the three results.

# Key Idea in Visual Programming



**Syntactically separate stream and non-stream arguments.**



Edit Params

Edit Parameters

order: 7

size: 128

direction: -1

OK Apply Close Cancel

## Higher-Order Functions

HOFs are functions that take functions as arguments and return functions. For example

$\text{map } f$

returns a function that applies function  $f$  to elements of a list. In a typical functional language (Haskell):

$\text{scan } (f, \text{init}, \perp) = \text{init}$

$\text{scan } (f, \text{init}, (x:xs)) = f(x, \text{scan}(f, \text{init}, xs))$

where  $\perp$  means no arguments. So,

$\text{scan } ((+), 0, \text{list})$

adds all the elements of a list.

## Higher-Order Functions



**Wan-Teh Chang**

**Edward A. Lee**

**Alan Kamas**

**Thomas M. Parks**

**UC Berkeley**

**Karim P. Khiar**

**Thomson SGS**