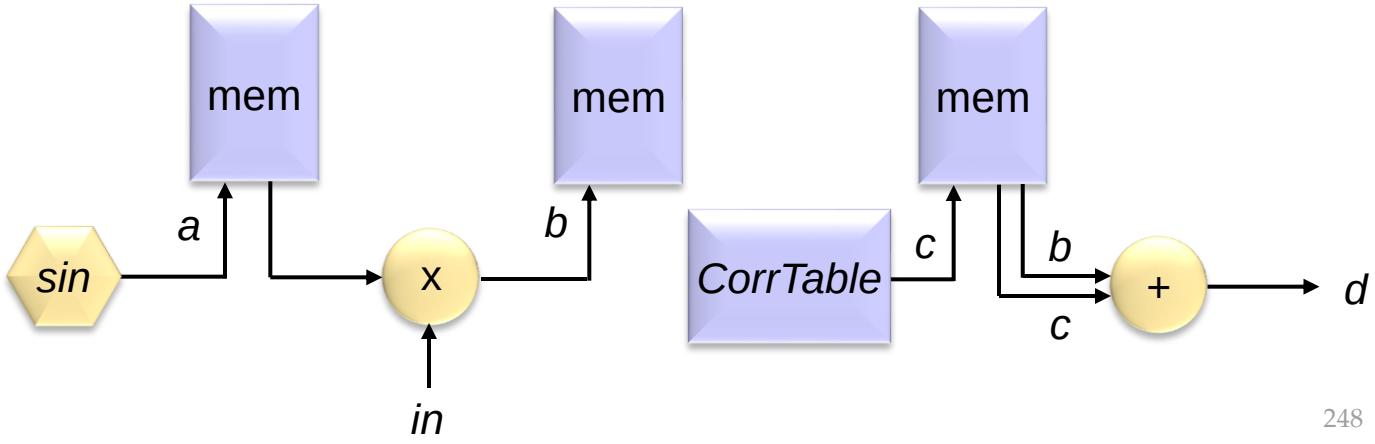
DRIVE THROUGH PROCESSING

"Drive thru" Data Processing

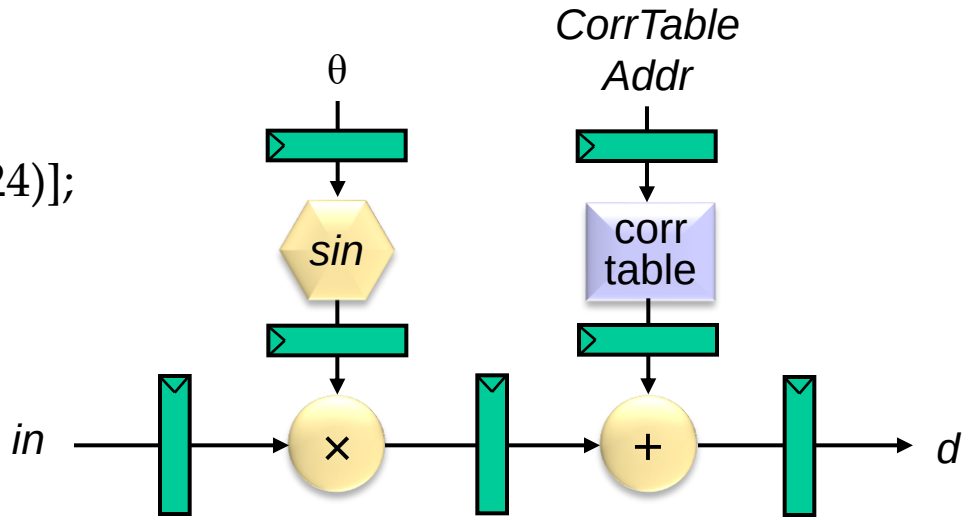
- Example implementation using a "standard high-level programming language"
  - Allocate memory space for temporary variables
  - Perform one complete task at a time and move data between buffers
  - Wastes lots of energy (power) on communication and memories
  - Ex: a = sin(1:1024);
    $b = in(1:1024) \ .* \ a(1:1024)$;
    c = CorrectionTableMem[addr(1:1024)];
    $d = b + c$;



"Drive thru" Data Processing

- Example implementation for an efficient real-time processing system
  - Process data as it flows by
  - Don't store any more data than is absolutely necessary
  - Don't request/generate data until exactly the cycle it is needed
  - Ex:

$a = \sin(\theta);$
$b = in * a;$
$c = \text{CorrTableMem}[addr(1:1024)];$
$d = b + c;$



"Drive thru" Data Processing

- I used to call this "Drive by" data processing—not a very nice name, but it does give a better sense of data flowing along and getting processed as it passes by
- Don't picture an In-N-Out Burger drive thru with cars lined up waiting
- Picture an automatic car wash where cars get pulled along at a *constant rate* and various steps are applied as cars pass by various stations:
  1) Soap applied
  2) Brushes
  3) Rinse
  4) Dry
- Multiple cars are cleaned at the same time which looks just like a pipelined datapath

"Drive thru" Data Processing

- The previous diagram illustrates the way data flows from input (*in* primarily) to the output (*d*) but it is not a valid pipelined block diagram