Large Multicore FFTs: Approaches to Optimization

Sharon Sacco and James Geraci

24 September 2008

This work is sponsored by the Department of the Air Force under Air Force contract FA8721-05-C-0002. Opinions, interpretations, conclusions and recommendations are those of the author and are not necessarily endorsed by the United States Government

Outline

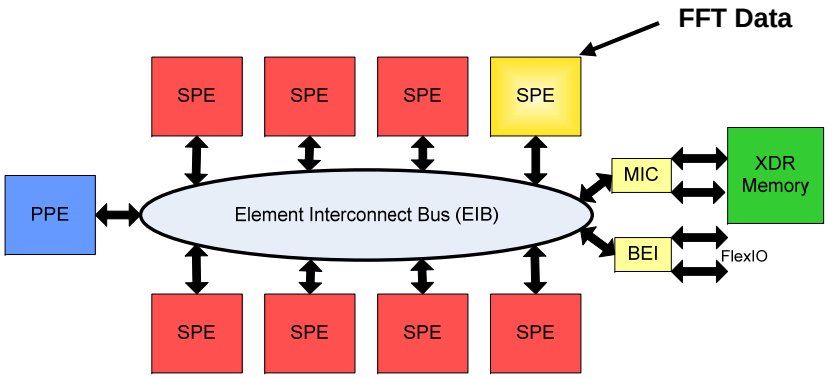
- **Introduction**
  - *1D Fourier Transform*
  - *Mapping 1D FFTs onto Cell*
  - *1D as 2D Traditional Approach*
- Technical Challenges
- Design
- Performance
- Summary

1D Fourier Transform

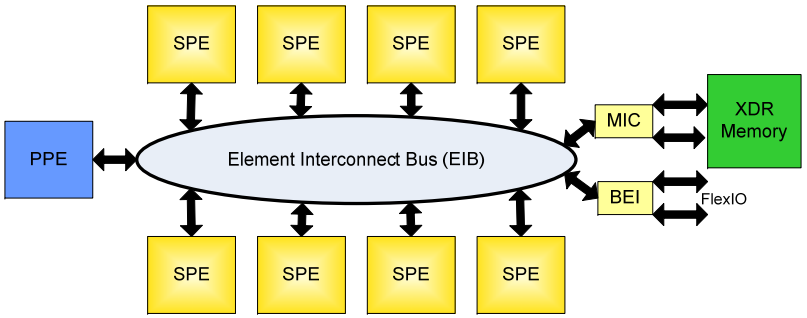
$$g_j = \sum_{k=0}^{N-1} f_k \, e^{-2\pi i jk/N}$$

- This is a simple equation
- A few people spend a lot of their careers trying to make it run fast

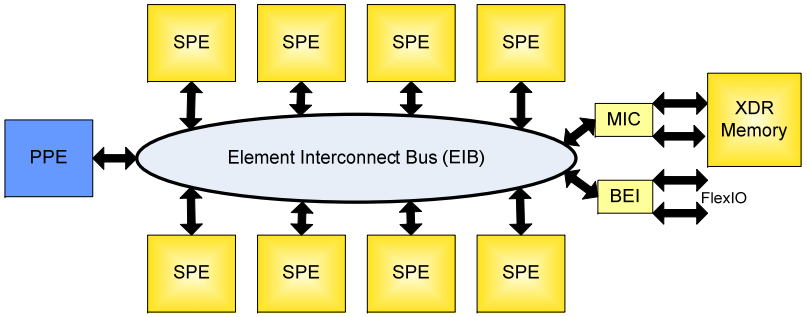
Mapping 1D FFT onto Cell



- **Small FFTs can fit into a single LS memory. 4096 is the largest size.**



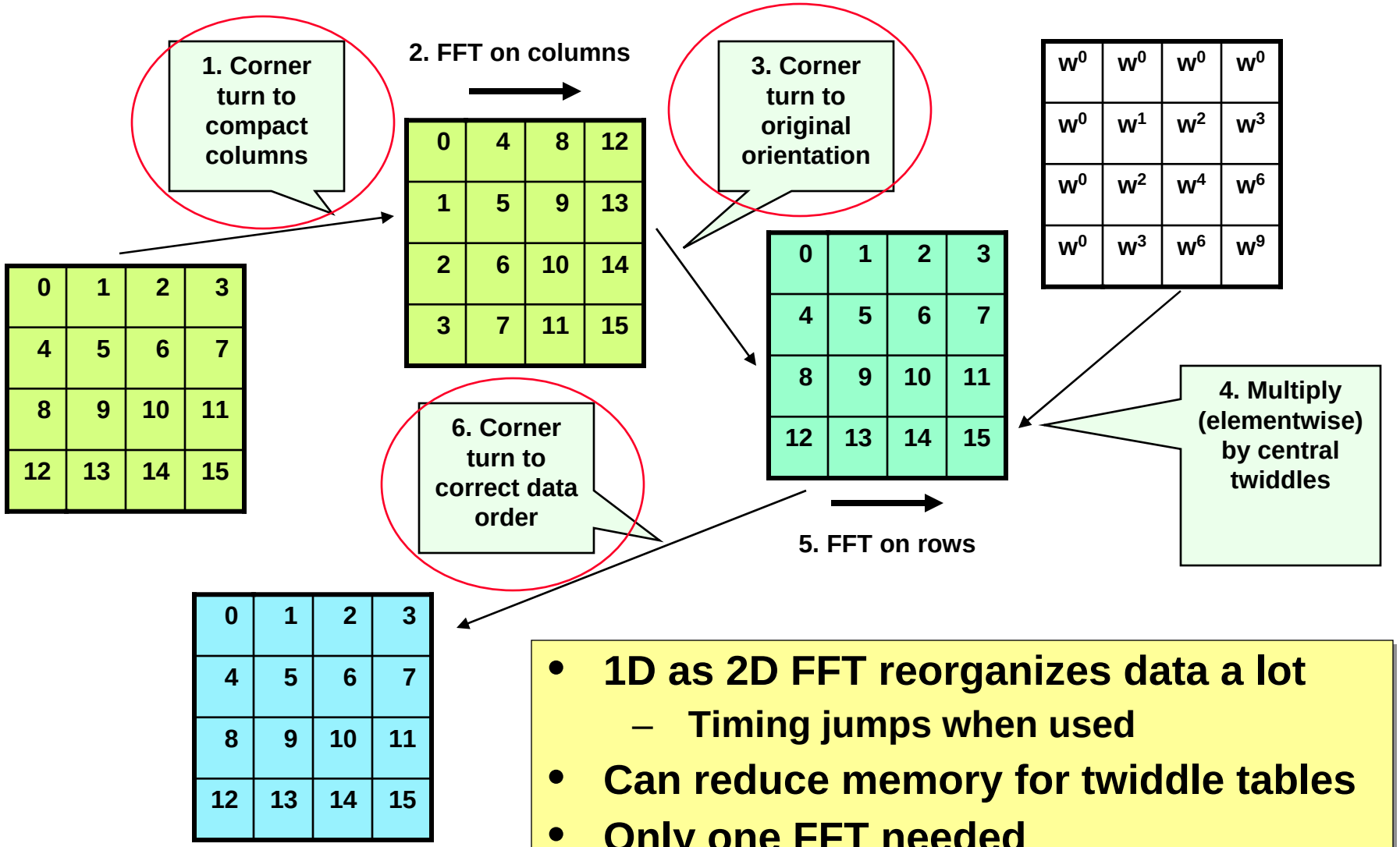
- **Medium FFTs can fit into multiple LS memory. 65536 is the largest size.**



- **Large FFTs must use XDR memory as well as LS memory.**

- **Cell FFTs can be classified by memory requirements**
- **Medium and large FFTs require careful memory transfers**

1D as 2D Traditional Approach

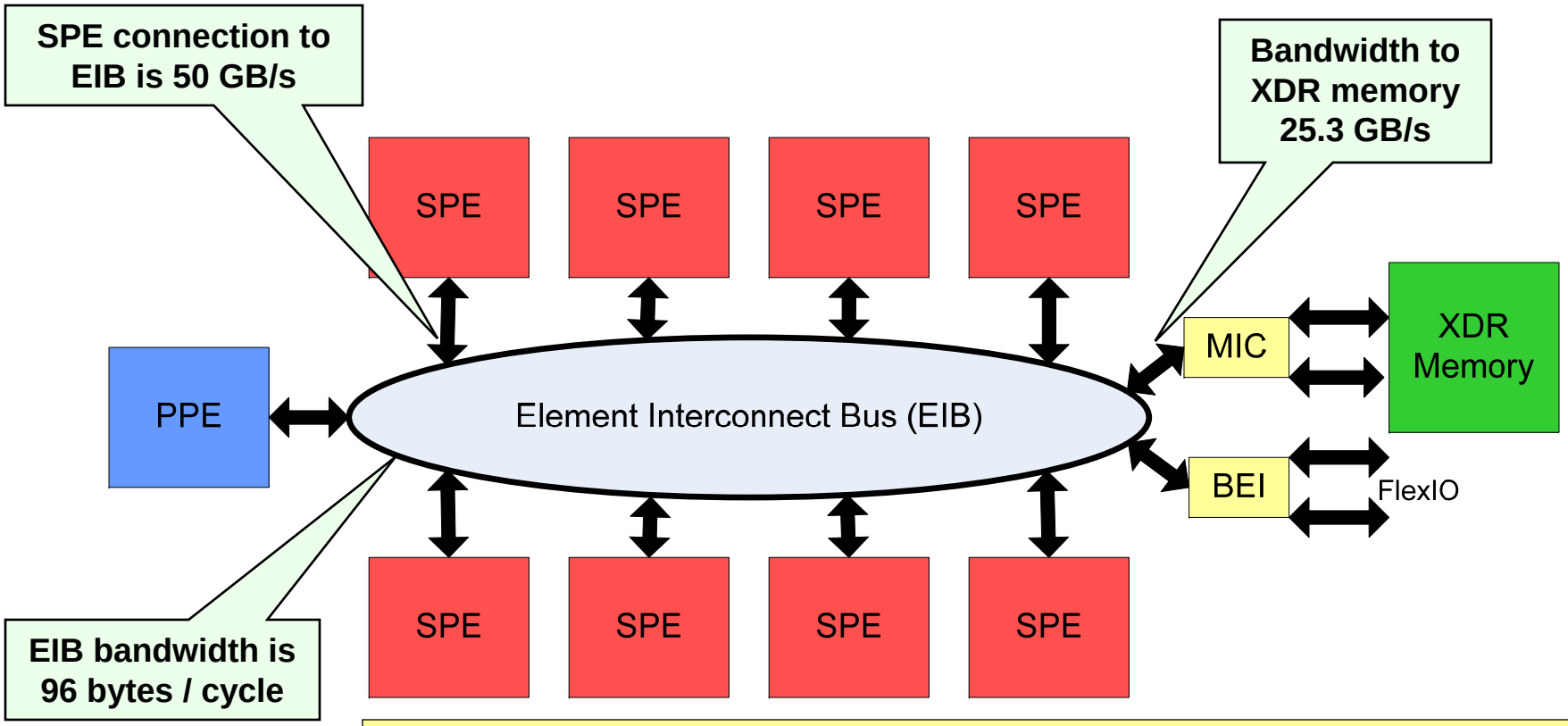


- **1D as 2D FFT reorganizes data a lot**
  - **Timing jumps when used**
- **Can reduce memory for twiddle tables**
- **Only one FFT needed**

Outline

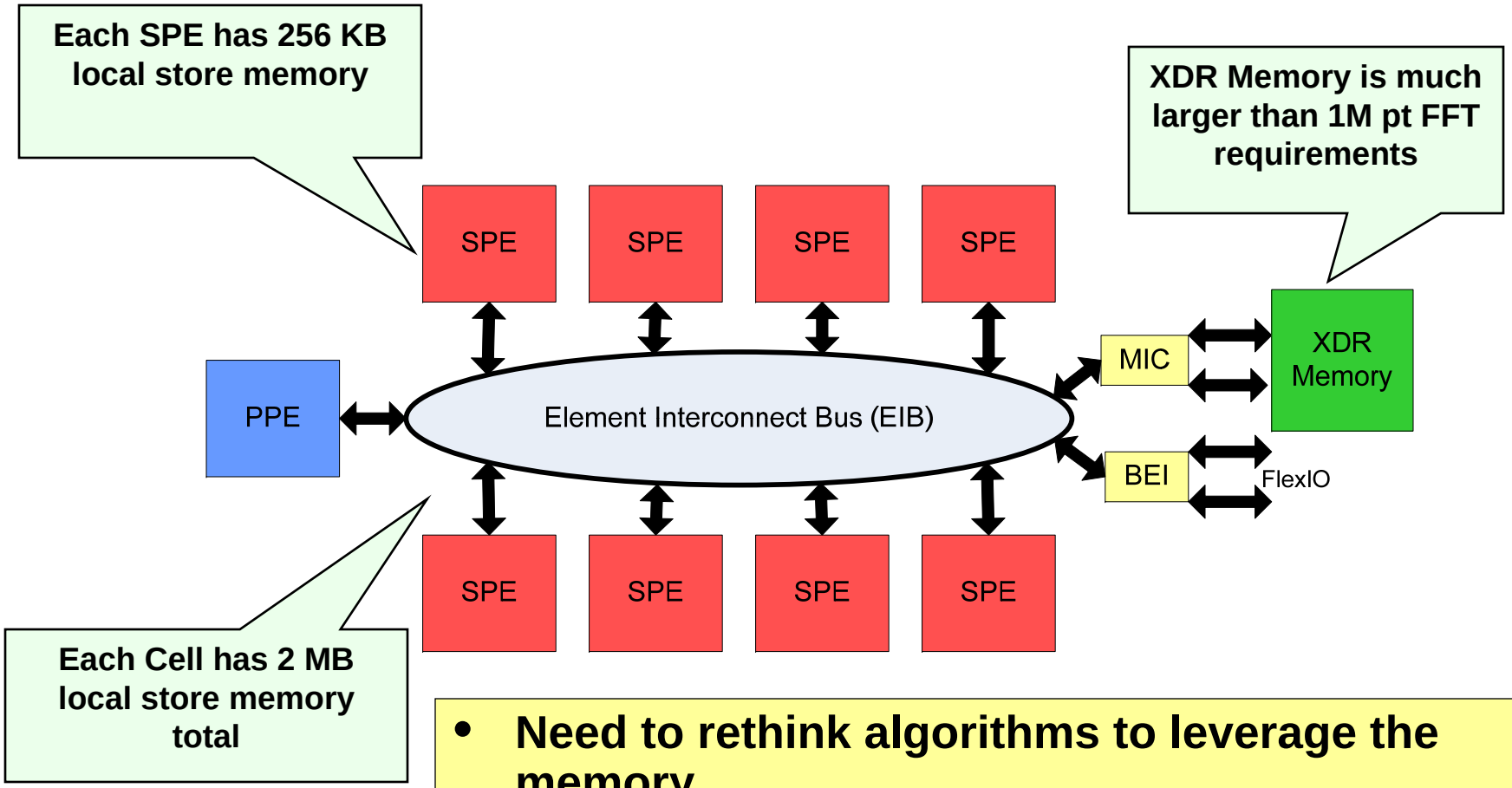
- Introduction
- **Technical Challenges**
  - ***Communications***
  - ***Memory***
  - ***Cell Rounding***
- Design
- Performance
- Summary

Communications



- **Minimizing XDR memory accesses is critical**
- **Leverage EIB**
- **Coordinating SPE communication is desirable**
  - **Need to know SPE relative geometry**

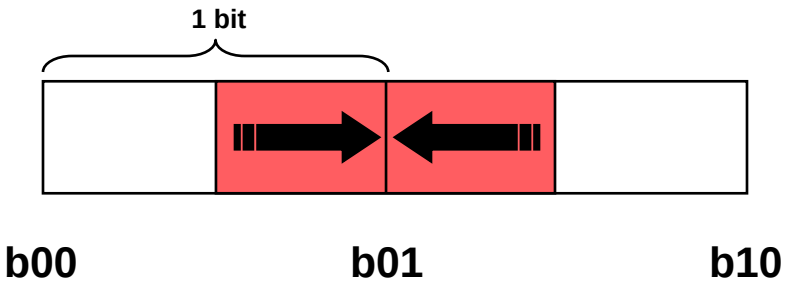
Memory



- **Need to rethink algorithms to leverage the memory**
  - **Consider local store both from individual and collective SPE point of view**

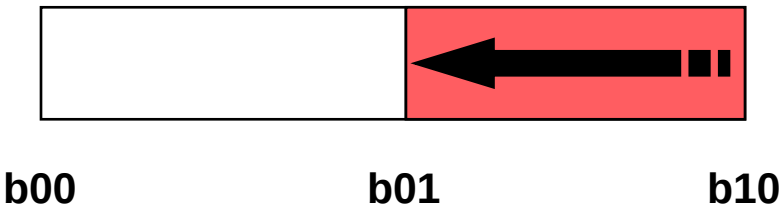
Cell Rounding

IEEE 754 Round to Nearest



- Average value – x01 + 0 bits

Cell (truncation)



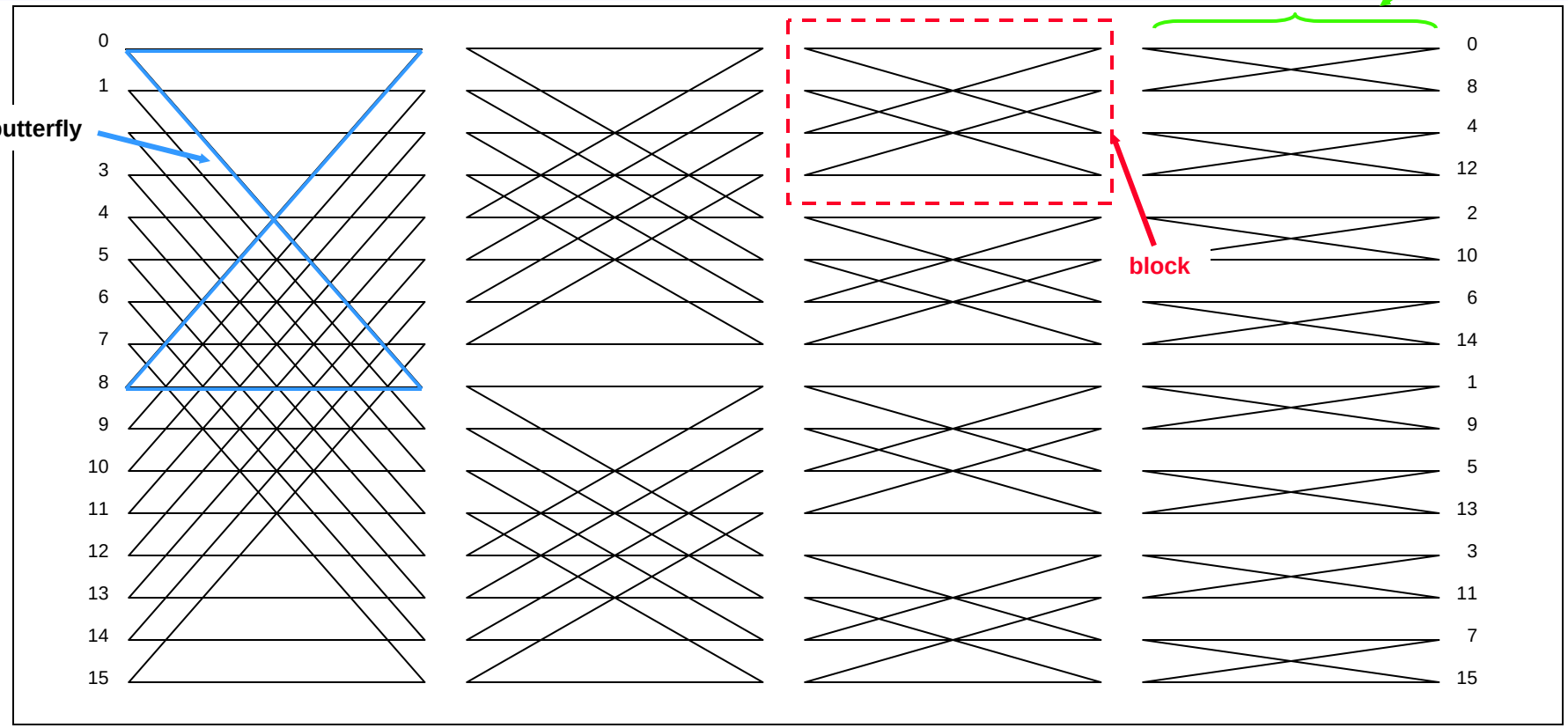
- Average value – x01 + .5 bit

- **The cost to correct basic binary operations, add, multiply, and subtract, is prohibitive**
- **Accuracy should be improved by minimizing steps to produce a result in algorithm**

Outline

- Introduction
- Technical Challenges
- **Design**
  - ***Using Memory Well***
    - ***Reducing Memory Accesses***
    - ***Distributing on SPEs***
    - ***Bit Reversal***
    - ***Complex Format***
  - *Computational Considerations*
- Performance
- Summary

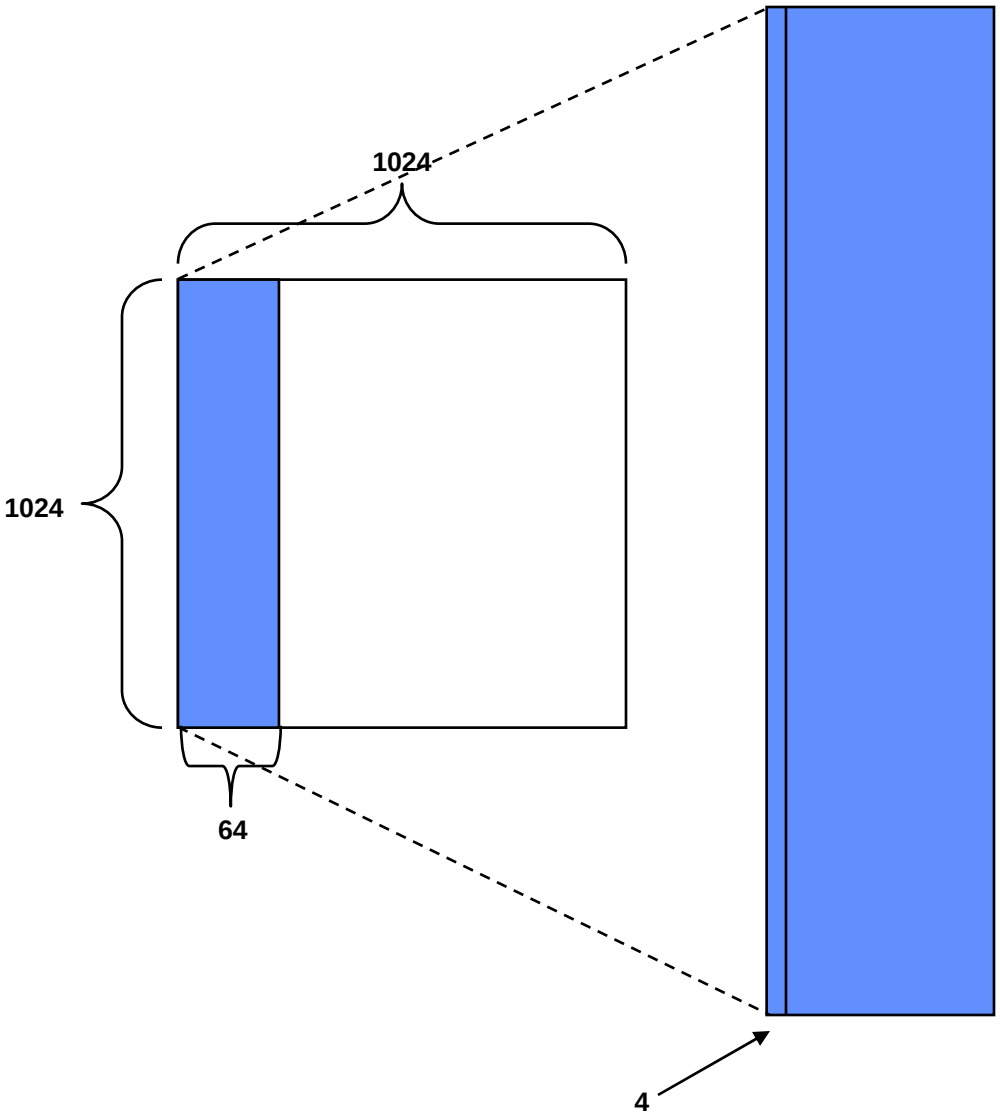
FFT Signal Flow Diagram and Terminology



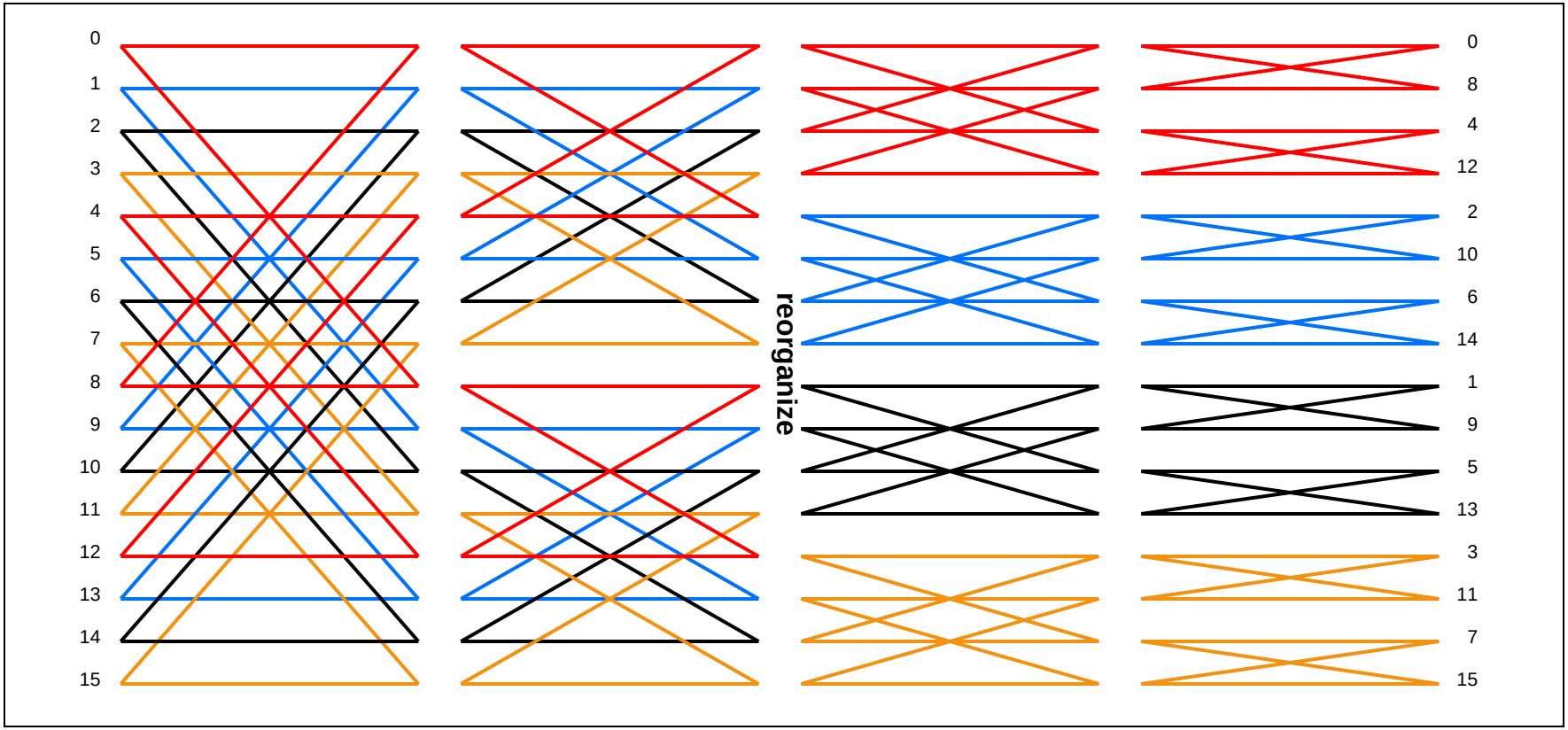
- **Size 16 can illustrate concepts for large FFTs**
  - **Ideas scale well and it is "drawable"**
- **This is the "decimation in frequency" data flow**
- **Where the weights are applied determines the algorithm**

Reducing Memory Accesses

- Columns will be loaded in strips that fit in the total Cell local store
- FFT algorithm processes 4 columns at a time to leverage SIMD registers
- Requires separate code from row FFTS
- Data reorganization requires SPE to SPE DMAs
- No bit reversal

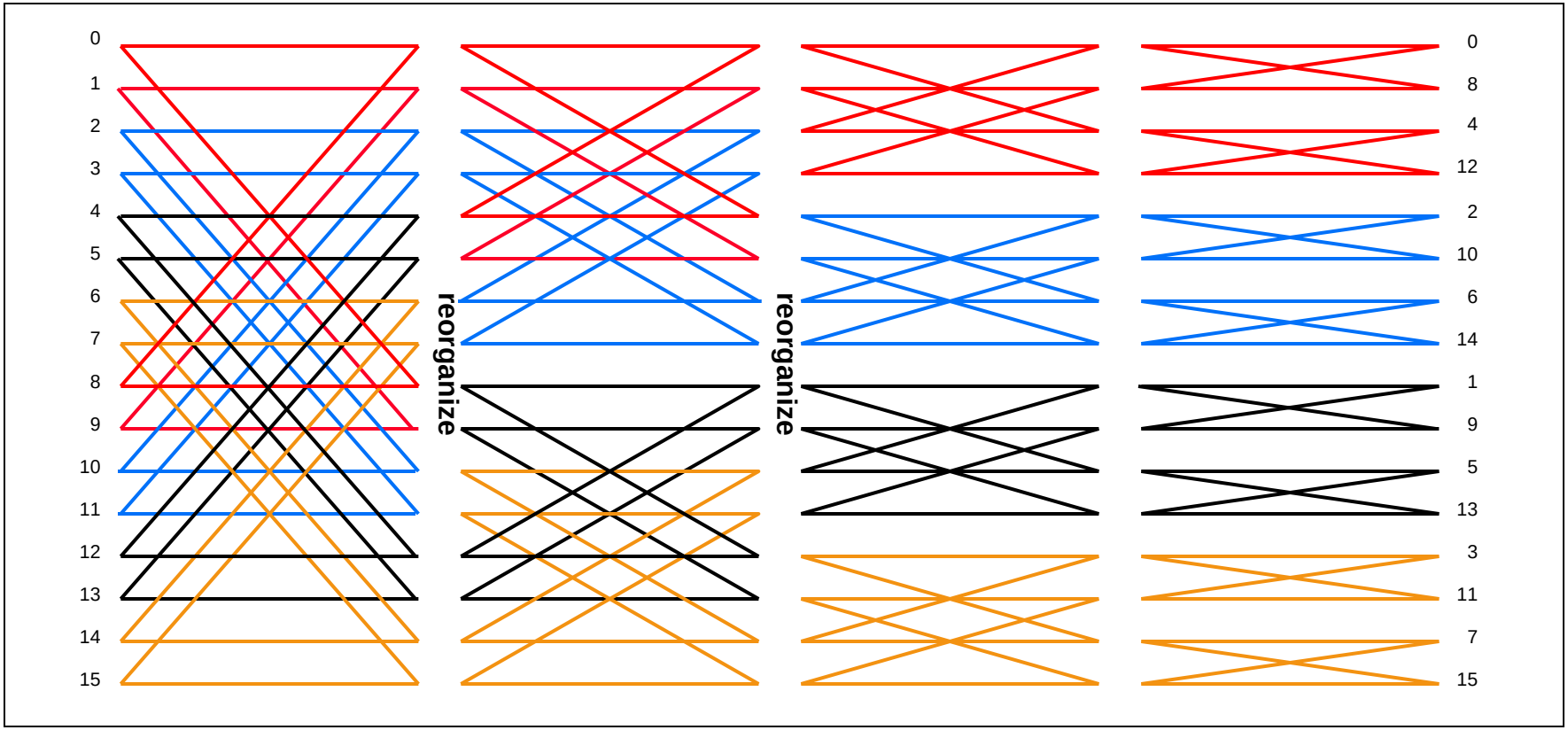


1D FFT Distribution with Single Reorganization



- **One approach is to load everything onto a single SPE to do the first part of the computation**
- **After a single reorganization each SPE owns an entire block and can complete the computations on its points**

1D FFT Distribution with Multiple Reorganizations



- **A second approach is to divide groups of contiguous butterflies among SPEs and reorganize after each stage until the SPEs own a full block**

Selecting the Preferred Reorganization

Typical N is 32k complex elements

Single Reorganization

- Number of exchanges
  $P * (P - 1)$
- Number of elements exchanged
  $N * (P - 1) / P$

Multiple Reorganizations

- Number of exchanges
  $P * \log_2 (P)$
- Number of elements exchanged
  $(N / 2) * \log_2 (P)$

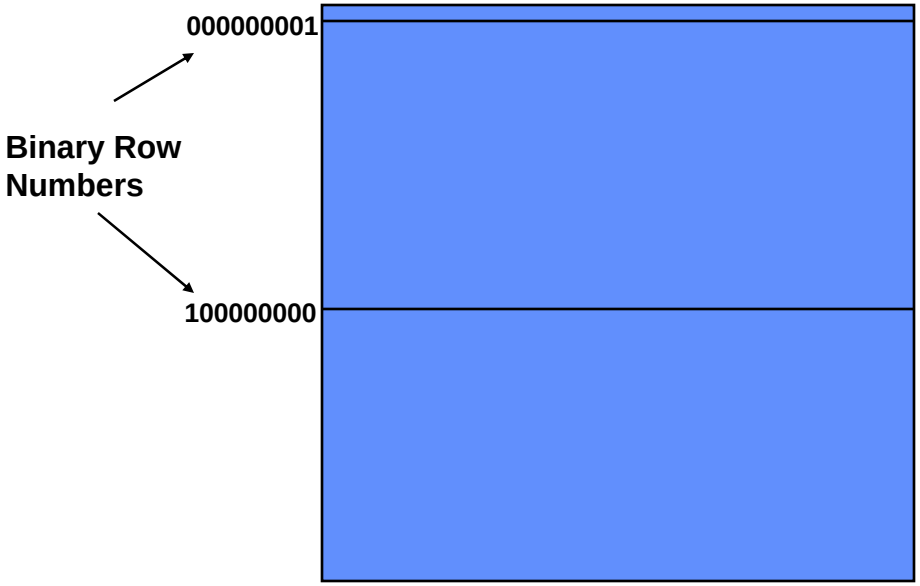
N - the number of elements in SPE memory, P - number of SPEs

Number of SPEs	Number of Exchanges	Data Moved in 1 DMA	Number of Exchanges	Data Moved in 1 DMA
2	2	N / 4	2	N / 4
4	12	N / 16	8	N / 8
8	56	N / 64	24	N / 16

- **Evaluation favors multiple reorganizations**
  - Fewer DMAs have less bus contention
    - Single Reorganization exceeds the number of busses
  - DMA overhead (~ .3μs) is minimized
  - Programming is simpler for multiple reorganizations

Column Bit Reversal

- Bit reversal of columns can be implemented by the order of processing rows and double buffering
- Reversal row pairs are both read into local store and then written to each others memory location



- Exchanging rows for bit reversal has a low cost
- DMA addresses are table driven
- Bit reversal table can be very small
- Row FFTs are conventional 1D FFTs

Complex Format

- Two common formats for complex
  - interleaved
  - split

- SIMD units need split format for complex arithmetic

- Complex format for user should be standard
- Internal format conversion is light weight
- Internal format should benefit the algorithm
  - Internal format is opaque to user

- **Interleaved complex format reduces number of DMAs**

Outline

- Introduction
- Technical Challenges
- **Design**
  - *Using Memory Well*
  - ***Computational Considerations***
    - ***Central Twiddles***
    - ***Algorithm Choice***
- Performance
- Summary

Central Twiddles

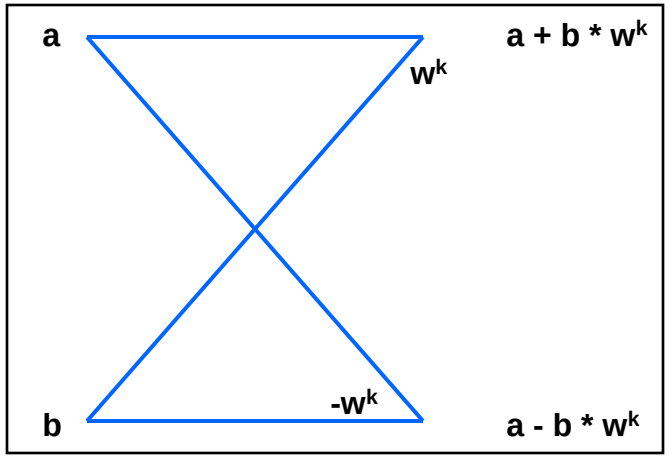
w^0	w^0	w^0	w^0	w^0	...
w^0	w^1	w^2	w^3		
w^0	w^2	w^4			
w^0	w^3		.		
w^0				.	
⋮					$w^{1023 * 1023}$

Central Twiddles for 1M FFT

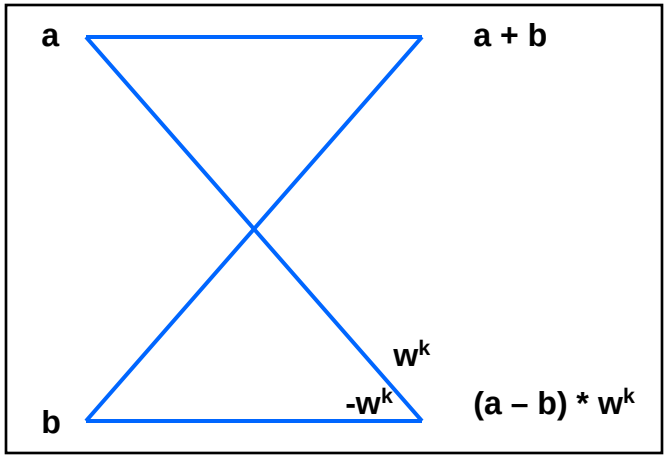
- **Central twiddles are a significant part of the design**

- **Central twiddles can take as much memory as the input data**
- **Reading from memory could increase FFT time up to 20%**
- **For 32-bit FFTs central twiddles can be computed as needed**
  - **Trigonometric identity methods require double precision**
    - **Next generation Cell should make this the method of choice**
  - **Direct sine and cosine algorithms are long**

Algorithm Choice



Cooley-Tukey



Gentleman-Sande

Computational Butterflies

- **Cooley-Tukey has a constant operation count**
  - **2 muladd to compute each result for each stage**
- **Gentleman-Sande varies widely in operation count**
  - **1 – 3 operations for each result**
- **DC term has the same accuracy on both**
- **Gentleman-Sande worst term has 50% more roundoff error when fused multiply-add is available**

Radix Choice

Cooley Tukey Radix 4

$t_1 = x_l * w^z$

$t_2 = x_k * w^{z/2}$

$t_3 = x_m * w^{3z/2}$

$s_0 = x_j + t_1$

$a_0 = x_j - t_1$

$s_1 = t_2 + t_3$

$a_1 = t_2 - t_3$

$y_j = s_0 + s_1$

$y_k = s_0 - s_1$

$y_l = a_0 - i * a_1$

$y_m = a_0 + i * a_1$

- **Number of operations for 1 radix 4 stage (real or imaginary: 9 (3 mul, 3 muladd, 3 add)**
- **Number of operations for 2 radix 2 stages (real or imaginary) : 6 (6 muladd)**
- **Higher radices reuse computations but do not reduce the amount of arithmetic needed for computation**
- **Fused multiply add instructions are more accurate than multiply followed by add**

- **Radix 2 will give the best accuracy**

- **What counts for accuracy is how many operations from the input to a particular result**

Outline

- Introduction
- Technical Challenges
- Design
- **Performance**
- Summary

Estimating Performance

I/O estimate:

- Number of byte transfers to/from XDR memory:
  33560192 (minimum)
- Bus speed to XDR: 25.3 GHz
- Estimated efficiency: 80%
- Minimum I/O time:
  1.7 ms

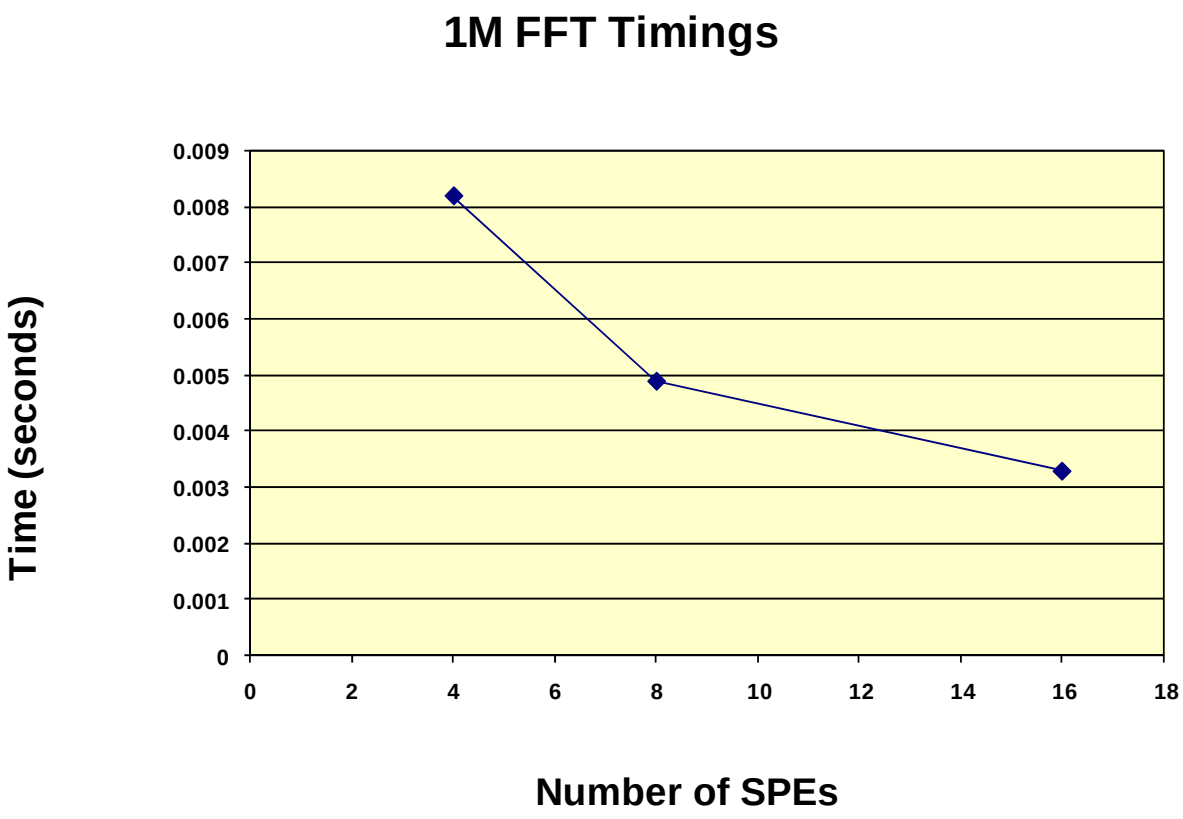
Computation estimate:

- Number of operations:
  104875600
- Maximum FLOPS: 205
- Estimated efficiency: 85%
- Minimum Computation time:
  .6 ms (8 SPEs)

- Timing estimates typically cannot include all factors
  - Computation is based on the minimum number of instructions to estimate
  - I/O timings are based on bus speed and amount of data
  - Experience is a guide for the efficiency of I/O and computations

1M FFT estimate (without full ordering): 2 ms

Preliminary Timing Results



- Timings were performed on Mercury CTES
  - QS21 @3.2 GHz Dual Cell Blades

- Timing results are close to predictions
  - 4 SPEs about a factor of 4 from prediction
  - 8 and 16 SPEs closer to prediction

Summary

- A good FFT design must consider the hardware features
  - Optimize memory accesses
  - Understand how different algorithms map to the hardware
- Design needs to be flexible in the approach
  - “One size fits all” isn’t always the best choice
  - Size will be a factor
- Estimates of minimum time should be based on the hardware characteristics

- 1M point FFT is difficult to write, but possible