



The Fast Fourier Transform on a Reconfigurable Processor

Gregory W. Donohoe

Institute for Advanced Microelectronics
at the
University of New Mexico

John Purviance

Colorado Springs, Co.

Pen-Shu Yeh

NASA Goddard Space Flight Center
Code 564



Overview



Reconfigurable Data Path Processor background

The Fast Fourier Transform

The Goertzel Algorithm

Fourier Transform Hyperspectral Imager data conversion

Performance

Conclusions



The Reconfigurable Data Path Processor



What is the RDPP?

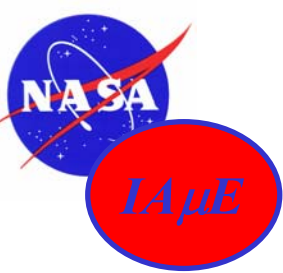


An embedded data processor VLSI chip for spacecraft.

- Targeted to rad-tolerant, 0.25 μ CMOS process.
- Implements a reconfigurable, synchronous data pipeline.
- Run-time reconfigurable.
- Serves as co-processor to a host CPU.
- Off-loads data intensive, streaming tasks from host.

Suite of support software.

- Application development, compiler, simulator, run-time.
- Integrated with existing software platforms.



Fusion of Technologies

IAμE VLSI Technologies

- Ultra-low power CMOS
- Radiation-tolerant CMOS

Radiation Tolerant
Low Power VLSI

Reconfigurable
Computing

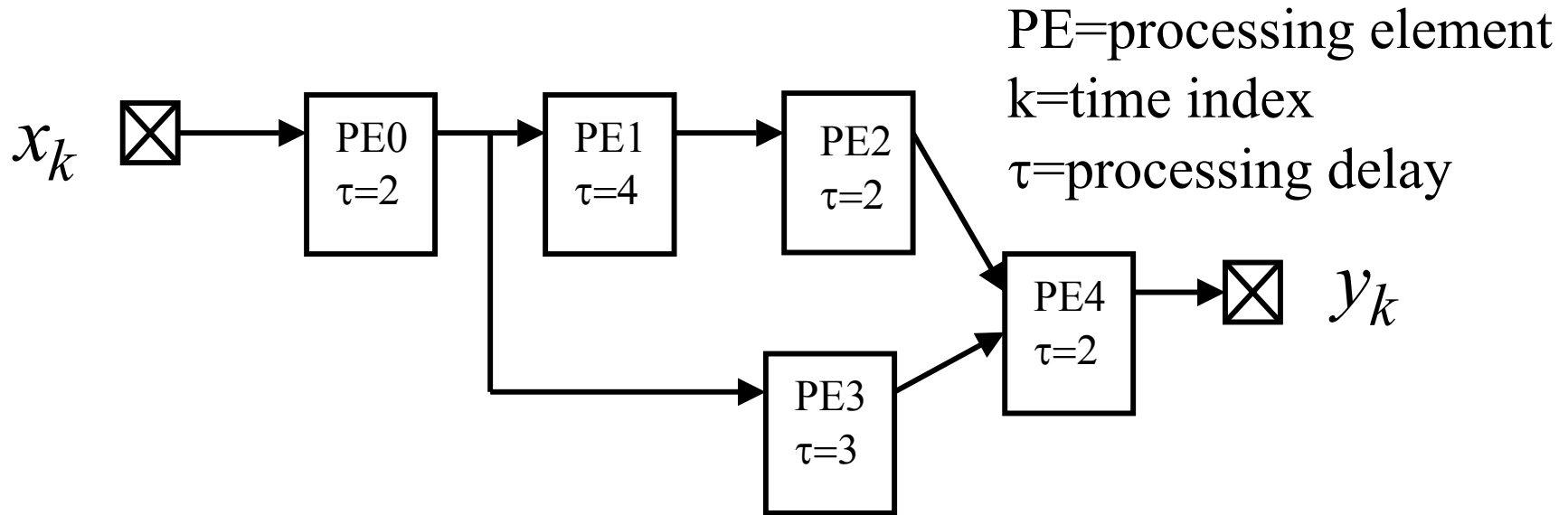
Software
Support

Reconfigurable Computing

- Architecture => throughput
- Reconfiguration => flexibility

Software support

- Dataflow model
- Leverage existing design environments



- Chain of processing elements (PEs)
- Blocking Read, Nonblocking Write.
- Conditional Data Path Selection instead of conditional branching.

Pipeline Execution Stages

1. Initialization: load data, “prime” the pipe.
2. Execution: process data stream in steady-state loop.
3. Termination: finish processing, flush the pipe.



RDPP Features



16 configurable on-board processing elements.

Each PE has

- Hardware Multiplier

- Arithmetic-logic unit

- Data path formatting logic

- Dynamic data path selection logic.

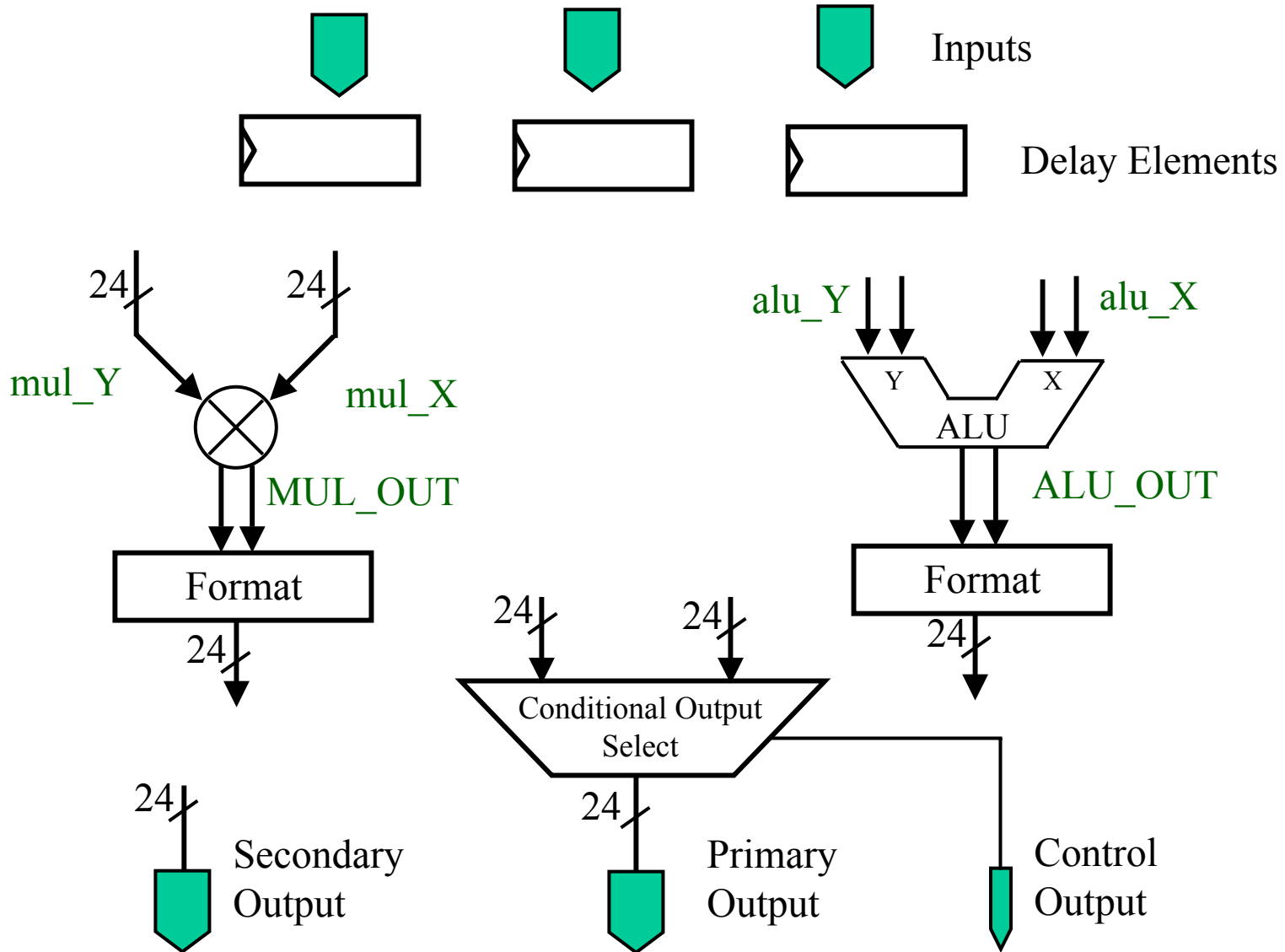
24-bit-wide data paths.

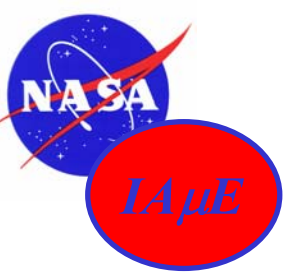
5 24-bit-wide, bidirectional I/O ports.

On-board program memory and execution unit.

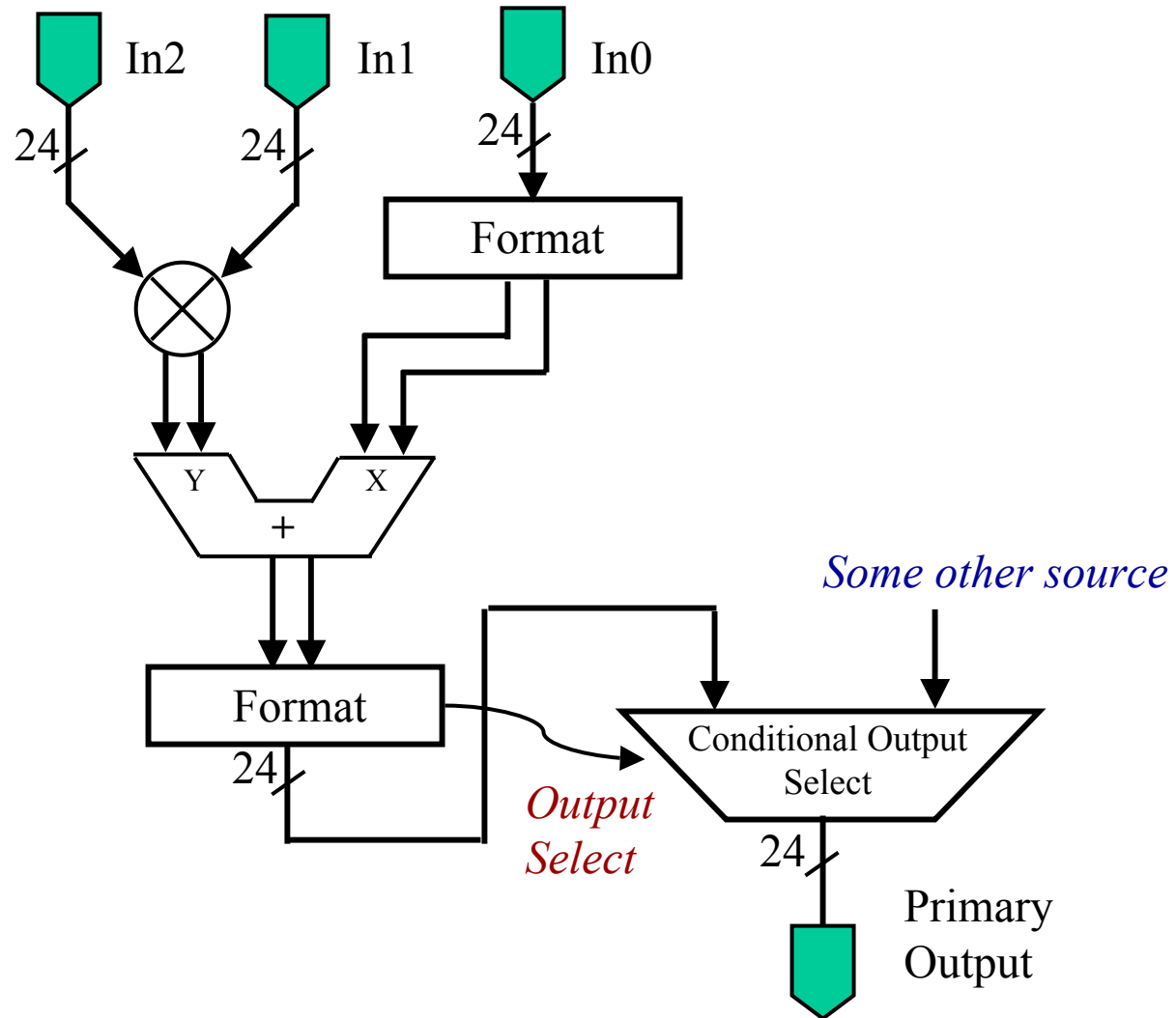
External control and synchronization signals.

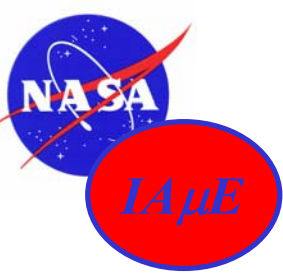
The RDPP acts as a computational accelerator for a host microprocessor.



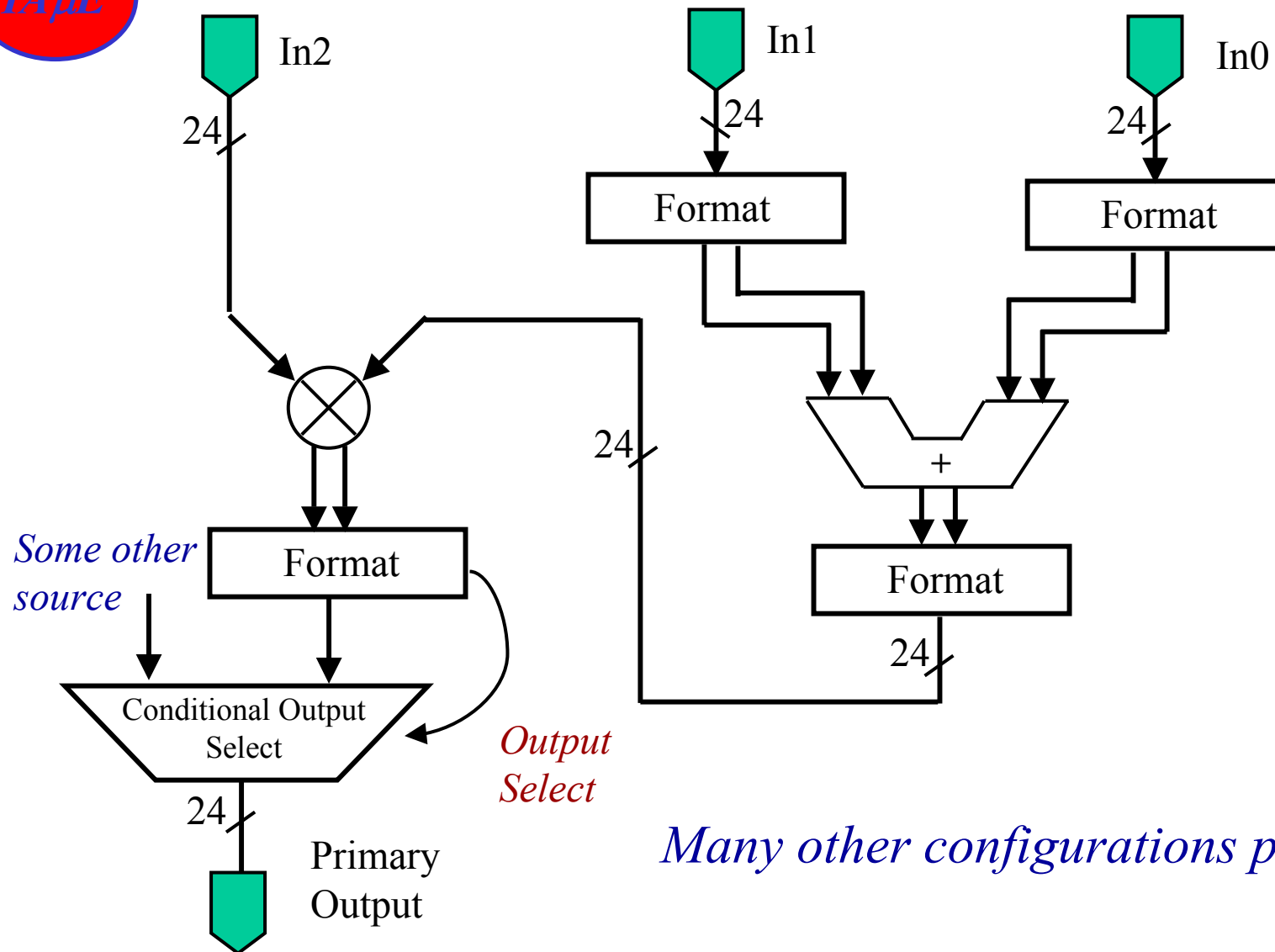


Scenario 1: Multiply-then-Add

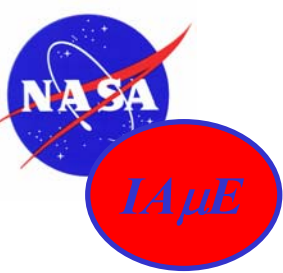




Scenario 2: Add-then-Multiply



Many other configurations possible.



Challenge Applications

Focal Plane Array Sensor Readout Correction

Nonuniformity correction

Hot pixel/dead pixel replacement

Fourier Transform Hyperspectral Imager data conversion

Appodization + Inverse Discrete Fourier Transform

Significantly reduces quantity of data to downlink.

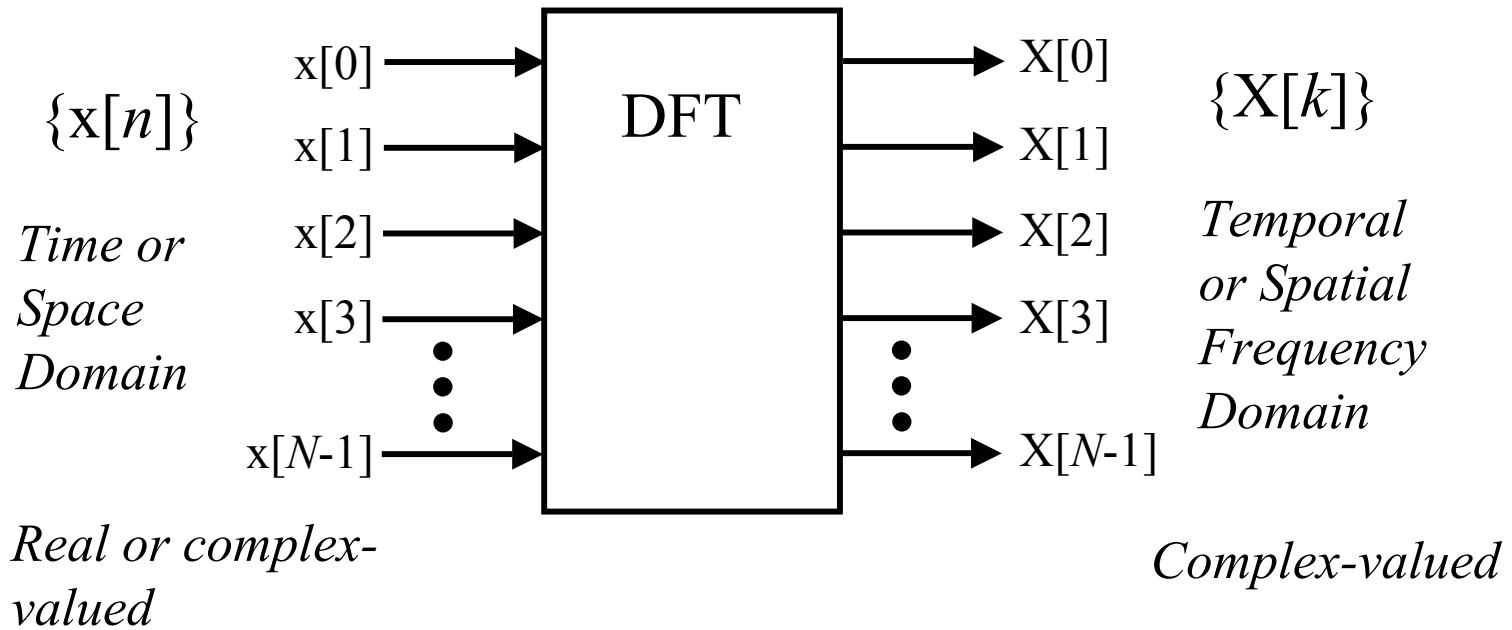
Important noise immunity advantages.



The Fast Fourier Transform

Discrete Fourier Transform

Maps an input sequence of N samples into an output sequence of N samples.



- $X[k]$, in polar form, is *amplitude* and *phase* (delay) at frequency k .
- $\{|X[k]|\}$ gives the *amplitude spectrum*.
- $\{X[k]^2\}$ gives the *power spectrum*.



Direct DFT Computation



$$X[k] = \sum_{n=0}^{N-1} x[n] W_N^{kn} \quad \begin{cases} n=0,1,\dots,N-1 \\ k=0,1,\dots,N-1 \end{cases}$$

$$W_N = e^{-j2\pi/N}$$

$$W_N^{kn} = e^{-j2\pi kn/N} = \cos(2\pi kn) - j \sin(2\pi kn)$$

This is the *basis set* for computing the DFT.

Direct Computation:

- $O(N^2)$ complex multiplies and accumulates (MACs).
- One complex multiply = four multiplies and two additions.
- One complex MAC is about four times as costly as a real MAC.
- On a sequential processor, this can be very slow.



Fast Fourier Algorithms



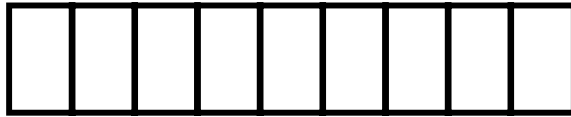
Fast Fourier Transform: any of several algorithms for computing the DFT quickly.

Cooley-Tukey FFT: Minimize the number of *multiply* operations.

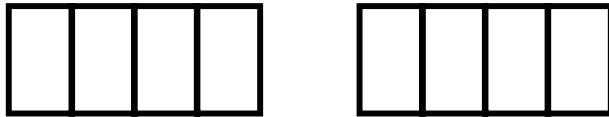
- Exploits symmetry and periodicity of the DFT basis set:
e.g. $\sin(\pi/4) = \sin(3\pi/4) = \sin(5\pi/4) = \sin(7\pi/4)$
- Divide-and-conquer algorithm.
- Computes DFT in $O(N \log N)$ MACs.

The Cooley-Tukey FFT algorithm makes
Fourier Analysis practical on desktop
workstations.

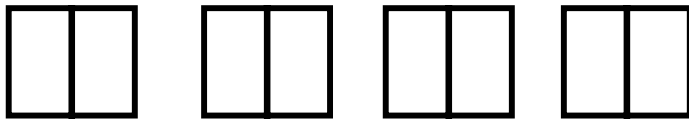
$$N = 8$$



$$N = \frac{8}{2} = 4$$



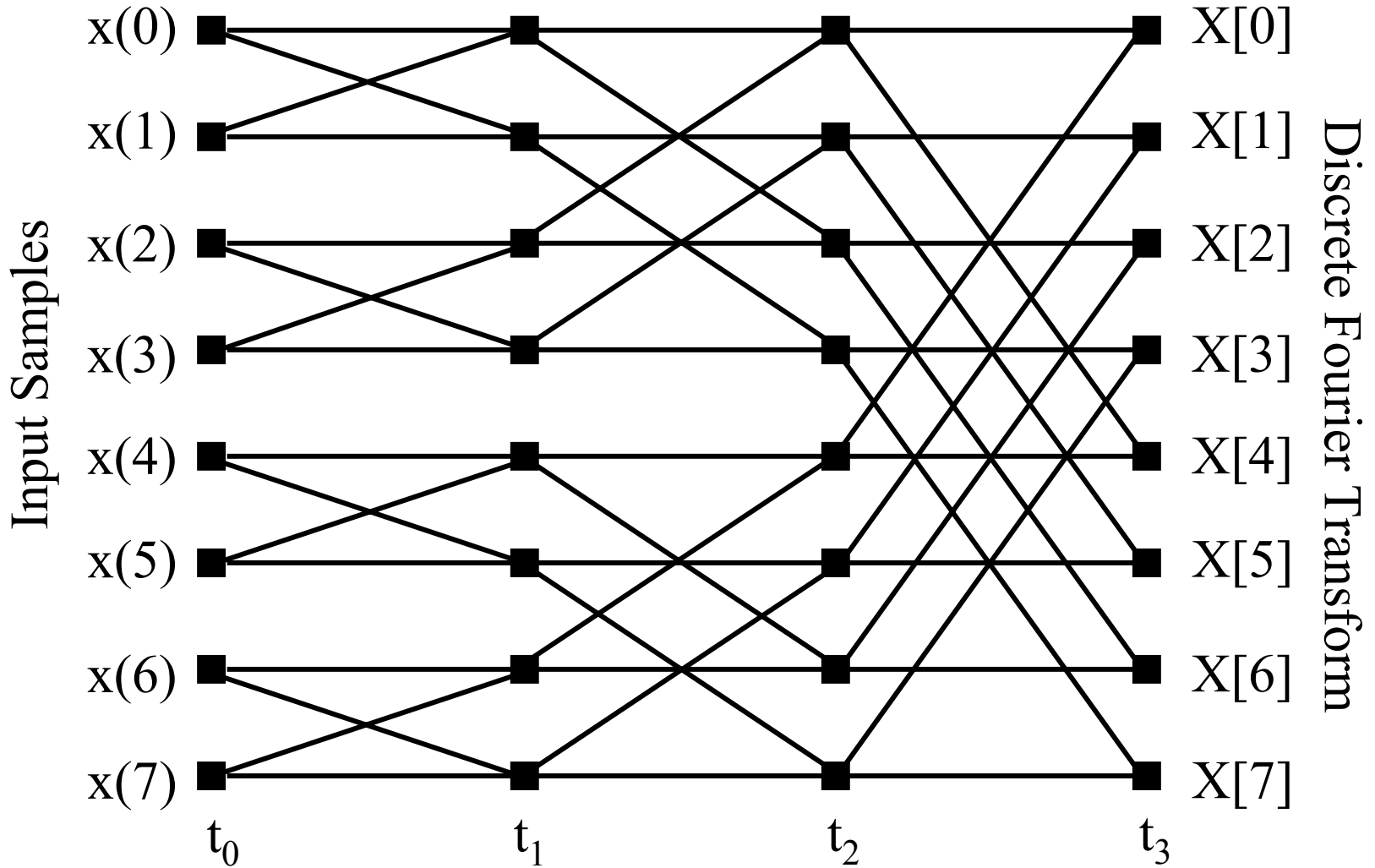
$$N = \frac{4}{2} = 2$$



1. Successively divide sequence in half.
2. When $N=2$, compute lowest level DFT.
3. Reconstruct back up the tree, re-using partial results to complete the DFT.

- Operates on blocks of $N=2^n$ samples.
- Entire sequence in memory at one time.
- Re-combination requires long-distance memory access.
- Algorithm of choice for sequential, random-access computers, where multiply-accumulate is critical operation.
- Difficult to parallelize into multi-processor machine.

FFT Signal Flow Diagram $N=8$





FFT on the RDPP



The Cooley-Tukey algorithm is optimized for random-access sequential-execution processors.

- ***Random access:*** all memory locations can be accessed in random order, in about the same amount of time.
- ***Sequential execution:*** One arithmetic-logic unit processes one instruction at a time.

The RDPP is neither random-access nor sequential-execution.

- Data enter and exit the processor in a stream.
- Multiple processing elements operate on the data stream simultaneously.

The Cooley-Tukey formulation of the FFT is not easy to implement in the RDPP.



Goertzel FFT Algorithm



Observe that: $W_N^{-kN} = 1$ where $W_N = e^{-j2\pi/N}$

$$\text{Then: } X[k] = \sum_{\ell=0}^{N-1} x[\ell] W_N^{-k(N-\ell)}$$

This has the form of a convolution, with impulse response:

$$H_k(z) = \frac{1}{1 - W_N^{-k} z^{-1}}$$

- Can be implemented as a recursive filter for each Fourier coefficient, run for N iterations.
- The Goertzel FFT is a bank of N of these filters, one per Fourier coefficient.
- Still requires $O(N^2)$ complex multiply-accumulates.

But wait, there's more...



Real-Valued Goertzel FFT



We can factor out the complex part

- Compute all but the last step with a real-valued 2-pole recursive filter
- One of the coefficients is -1 , eliminating a multiply.

Define intermediate variable $v_k[n]$:

$$v_k[n] = x[n] + 2 \cos(2\pi k / n) v_k[n-1] - v_k[n-2]$$

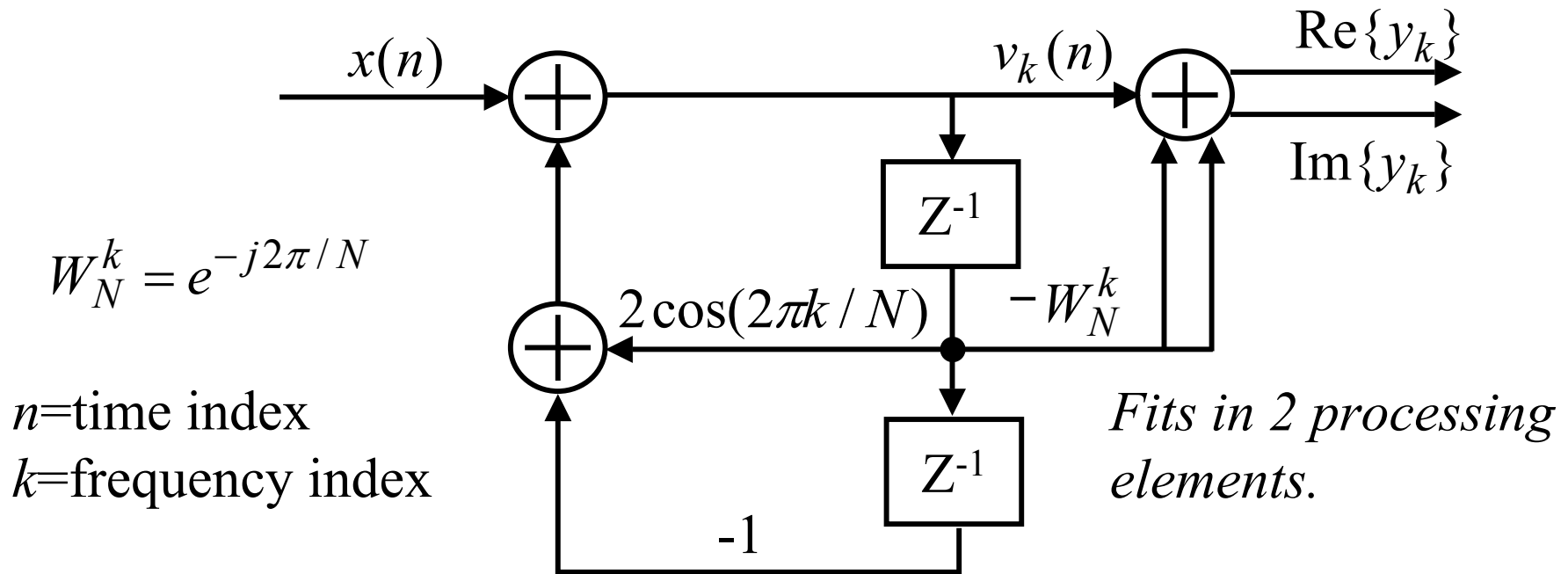
If we need the complex result, we include it in one final step:

$$X[k] = v_k[N] - W_N^k v_k[N-1]$$

If we only need the magnitude, then:

$$|X[k]| = v_k[N] - v_k[N-1]$$

Goertzel Signal Flow

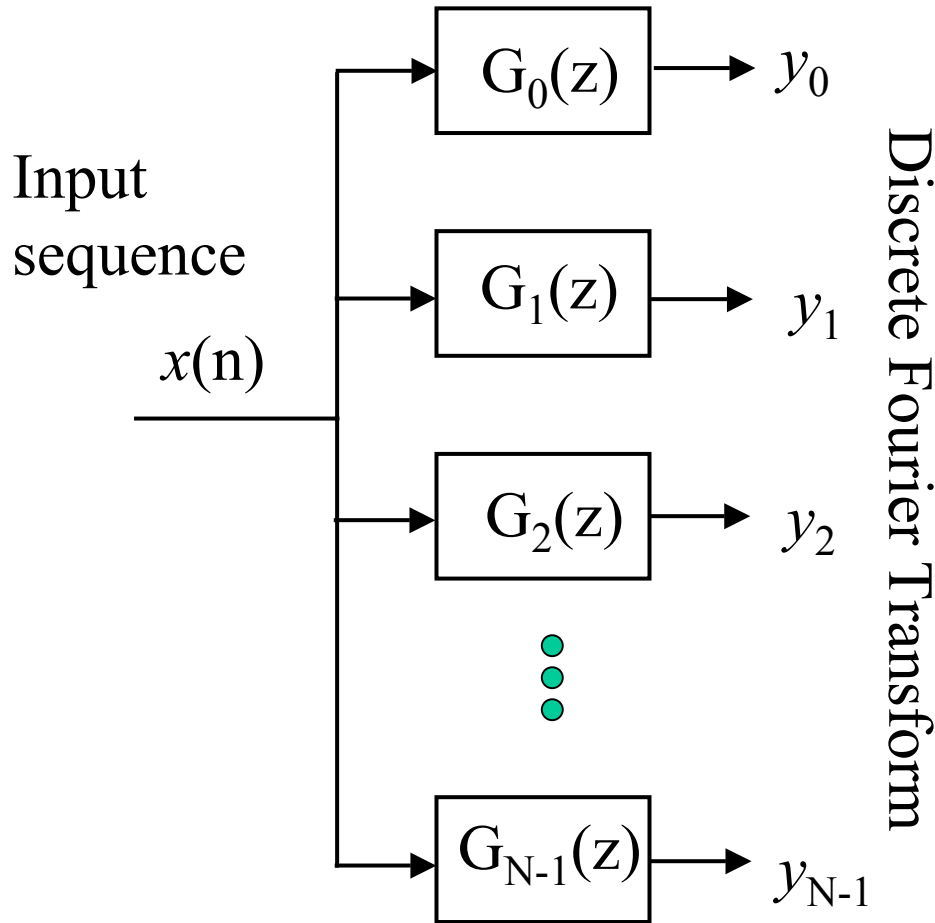


$$H_k(z) = \frac{1 - W_N^{-k} z^{-1}}{1 - 2 \cos(2\pi k / N) z^{-1} + z^{-2}}$$

$$v_k(n) = 2 \cos(2\pi k / N) v_k(n-1) - v_k(n-2) + x(n)$$

$$y_k = v_k(N) - W_N^k v_k(N-1)$$

- Run until all input samples are processed.
- Then multiply by W_N^k



DFT points are computed separately, in parallel.

Output is complex-valued.

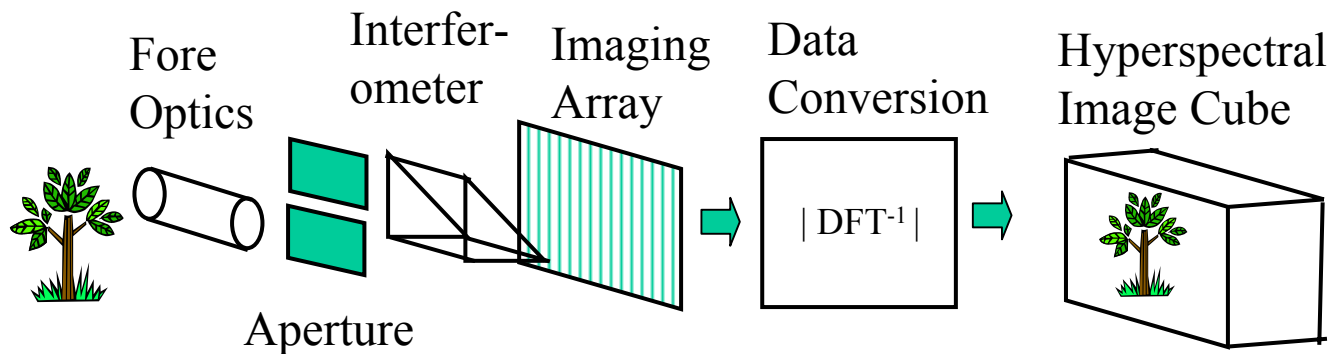
Each $G(z)$ filter processes every input sample.



Algorithm Comparison



Cooley-Tukey	Goertzel
$O(N \log N)$ complex multiply-accumulates	$O(N^2)$ real multiply-accumulates.
Block computation.	Point computation.
N must be a composite number ($N=2^m$).	N is arbitrary.
Butterfly network.	Small recursive loop.
Complex, multi-step data dependencies.	Coefficients computed independently
Difficult to parallelize.	Easy to parallelize.



*Data arrive as a Fourier Transform of spectral response.
Conversion requires inverse FFT.*

Why on-board data conversion?

1. Noise immunity.

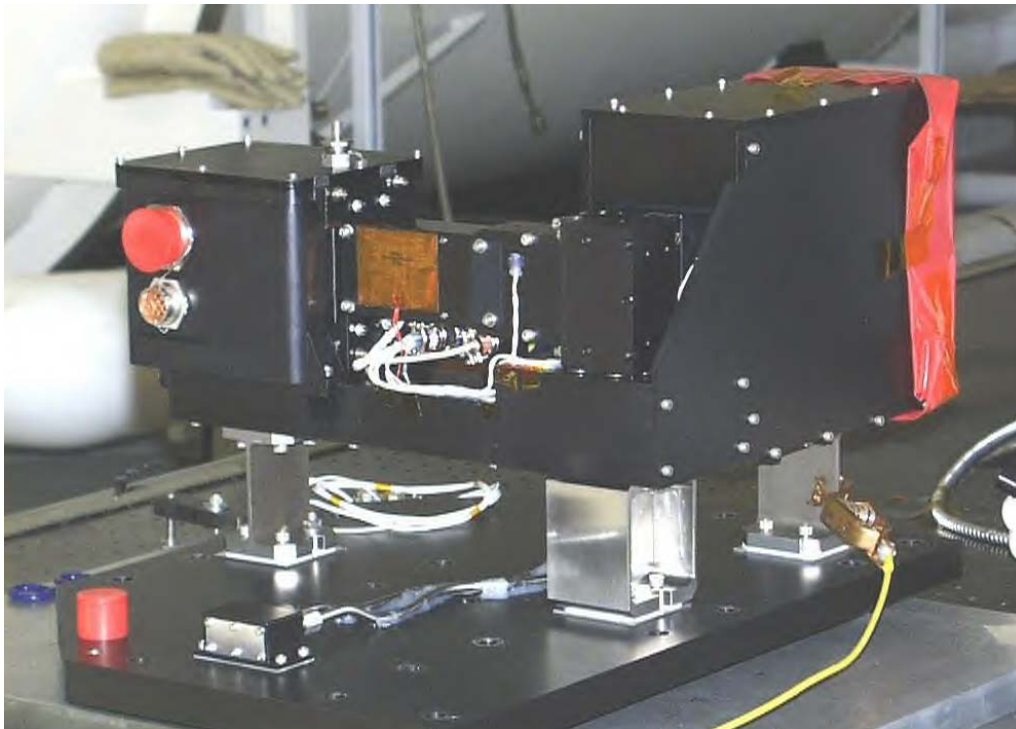
Inverse transform highly sensitive to transmission noise.

2. Data compression

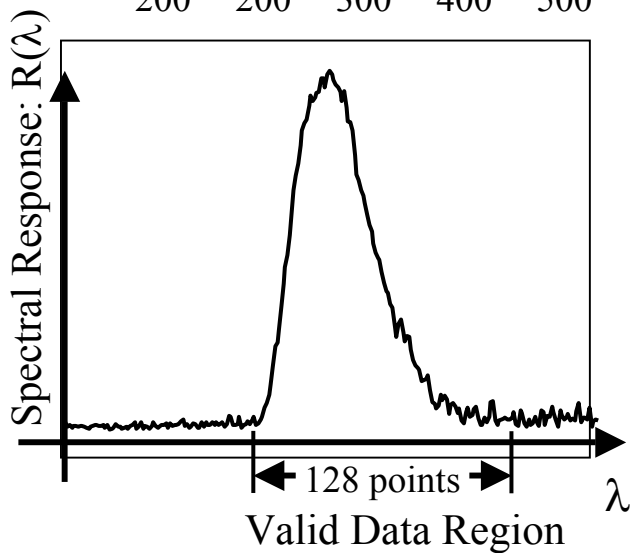
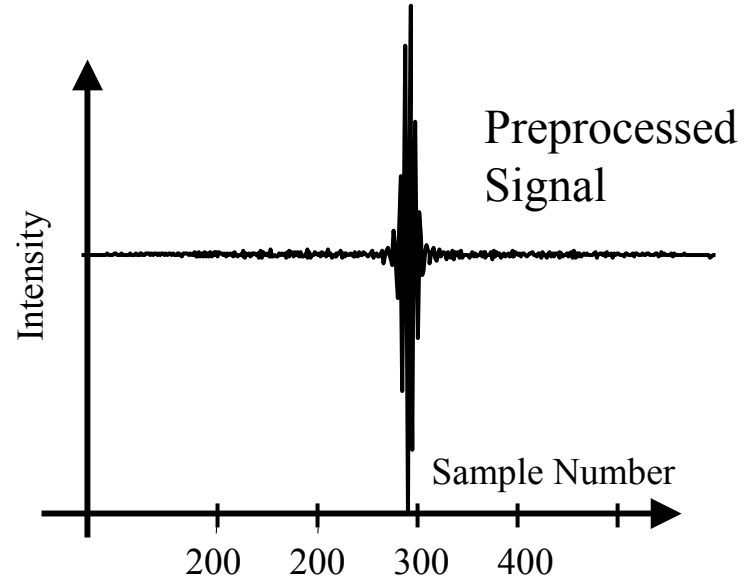
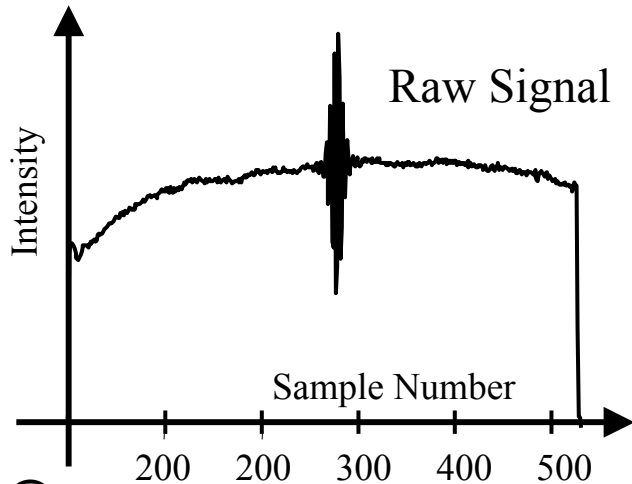
Converted signal carries considerably less data.

Flown on Air Force MightySat II.1 Satellite

- Launched July 19, 2000
- AFRL Space Vehicles Directorate proof-of-concept “lab bench”

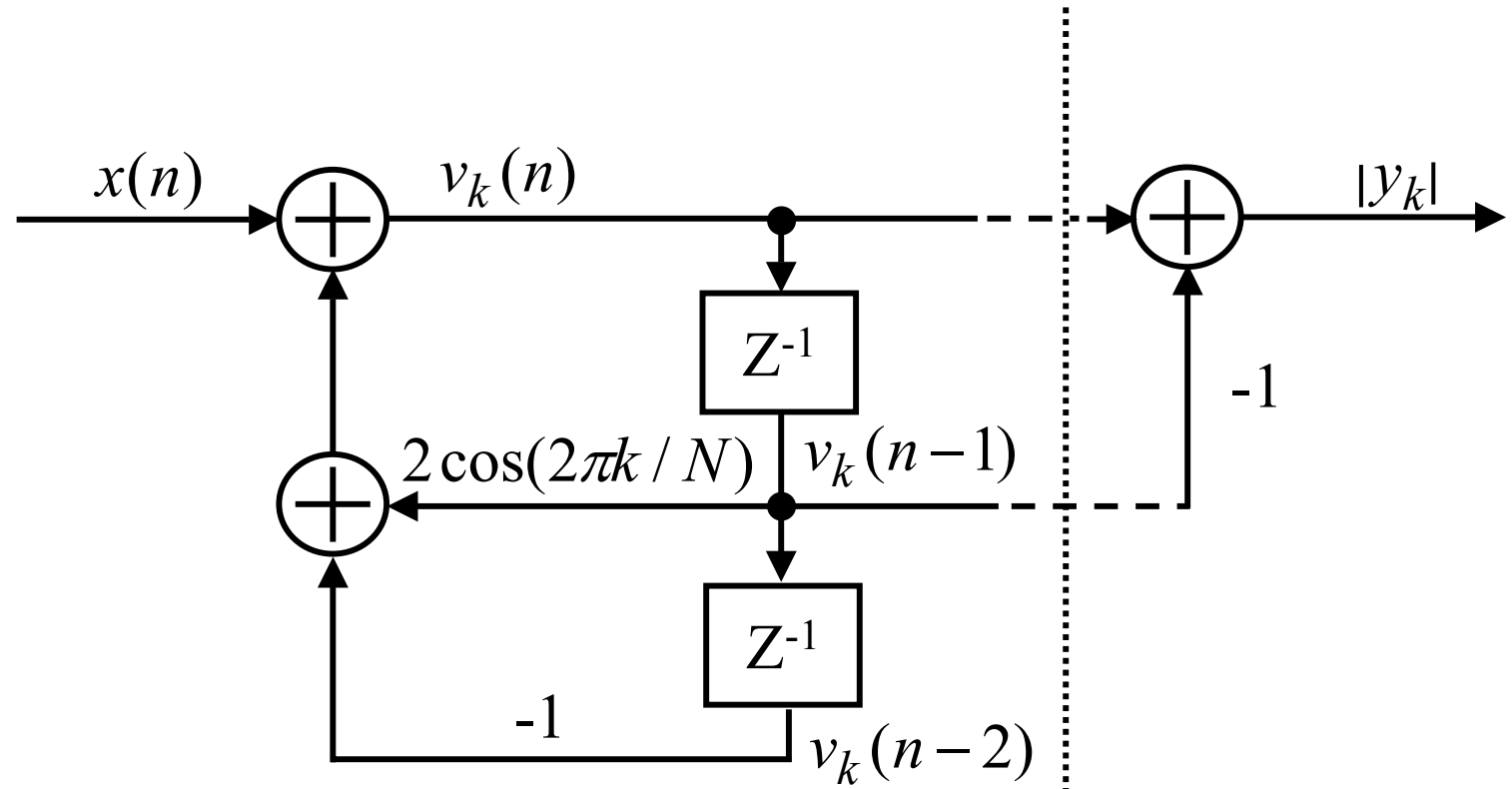


- Built by Kestrel Corporation.
- 470-1050 nm spectral range
- One picture every 3 days
- On-board data conversion experiment did not function.



Converted signal

- Appodized, zero-mean
- Hamming window
- Inverse FFT
- Cropped



128 iterations of recursive filter loop

Subtract last two samples

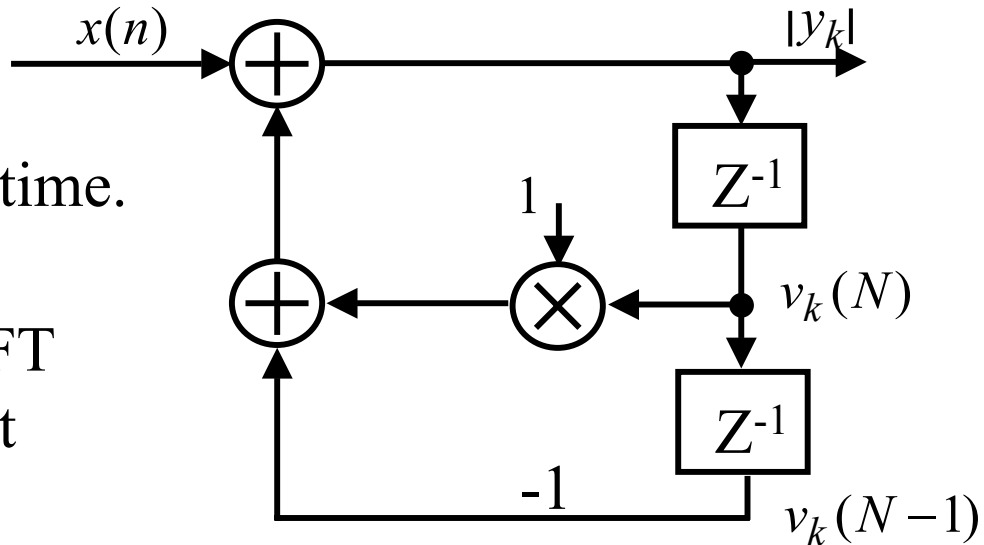
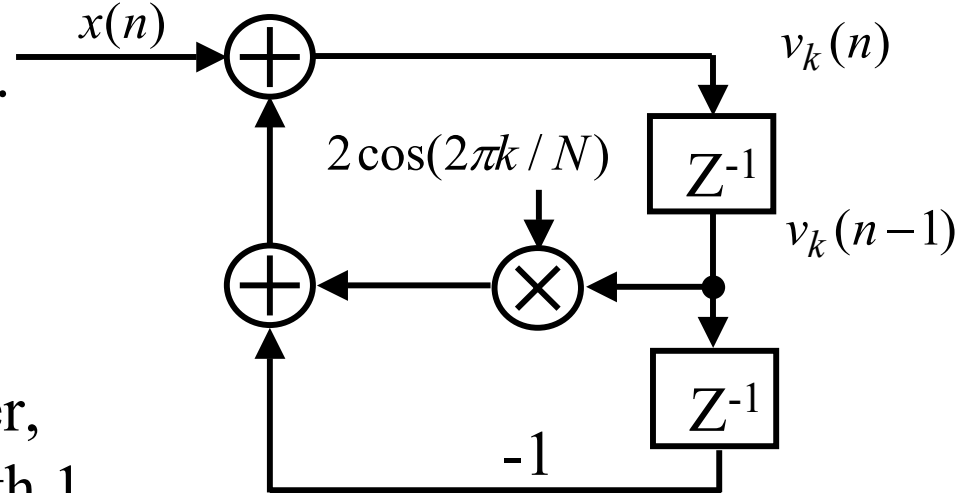
1. Run this loop 127 times.

2. Halt the RDPP.

3. Re-load constant register, replacing $2 \cos(2\pi k/N)$ with 1.

4. Run the loop one more time.

5. Merge the individual FFT coefficients into the output stream.





Data Scaling



The RDPP does not support floating point arithmetic.
Floating-point processing elements would be too large and slow.

RDPP has fixed-point data paths:

- 48 bits within PEs
- 24 bits between PEs.

Data must be scaled to deal with possible overflow or saturation.

- Data path scaling logic is provided in each PE.
- Scaling is set up at configuration time.

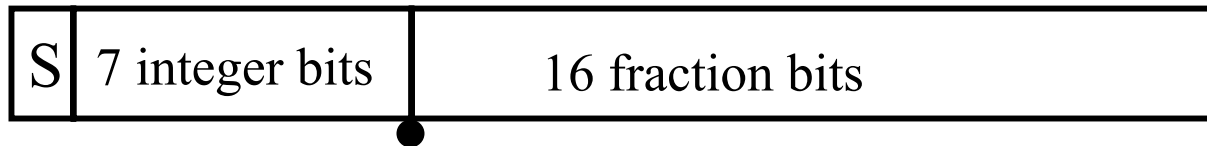


Scaling the FTESI Data

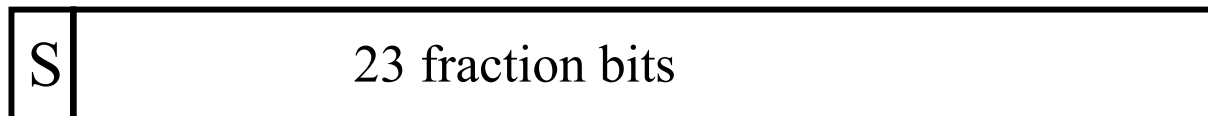


Assume 16-bit input data
(The Kestrel imager captures 12 bits)

- Scale input data so that $|x_k| < 1$
- Use this format for input data:



- Scale filter coefficients so that $|b_k| < 1$
- Use this format:

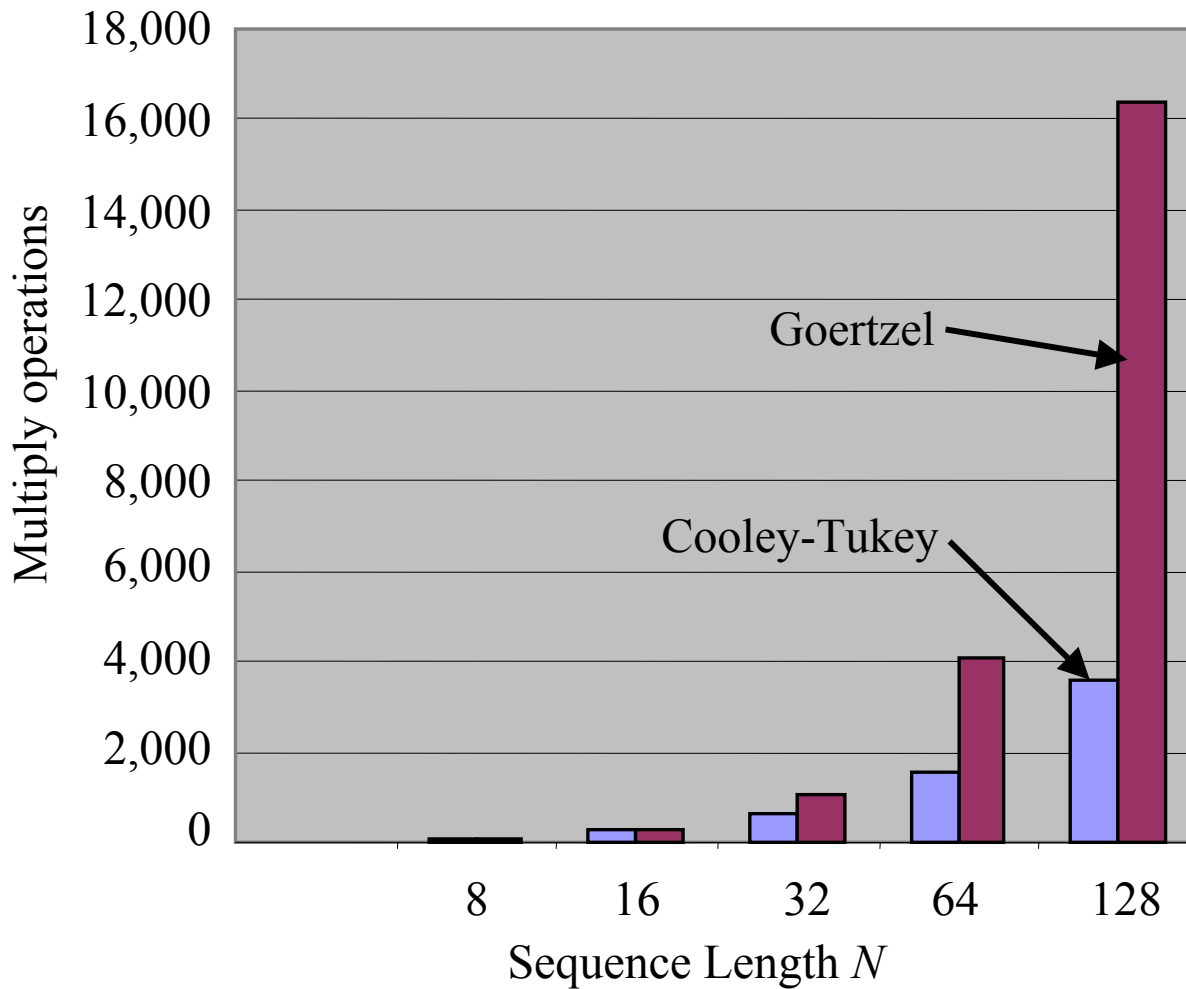


- We can compute $|X_k|$ without overflow, with 23 bits of precision.



Performance

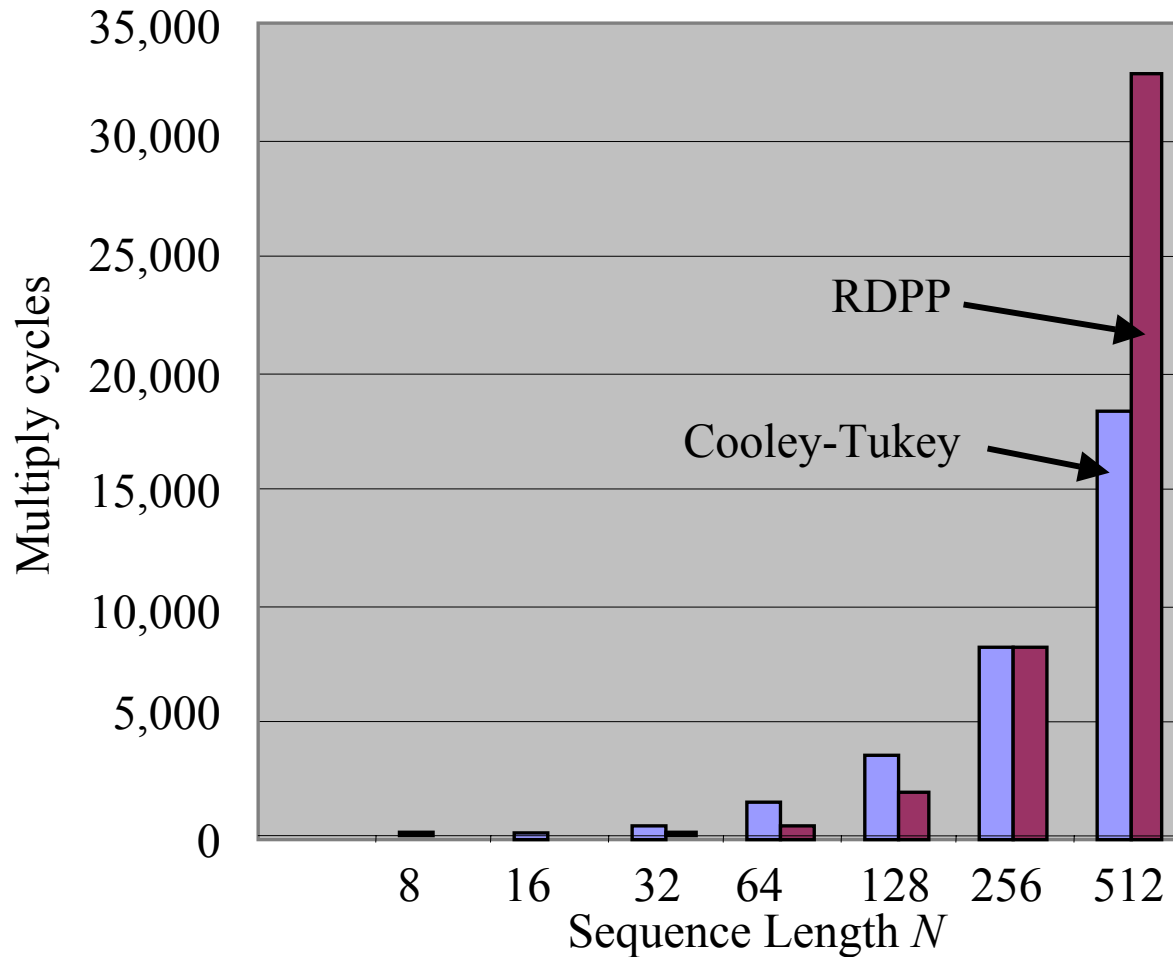
Multiply Count



For $N > 16$, Goertzel Algorithm requires more multiplies.

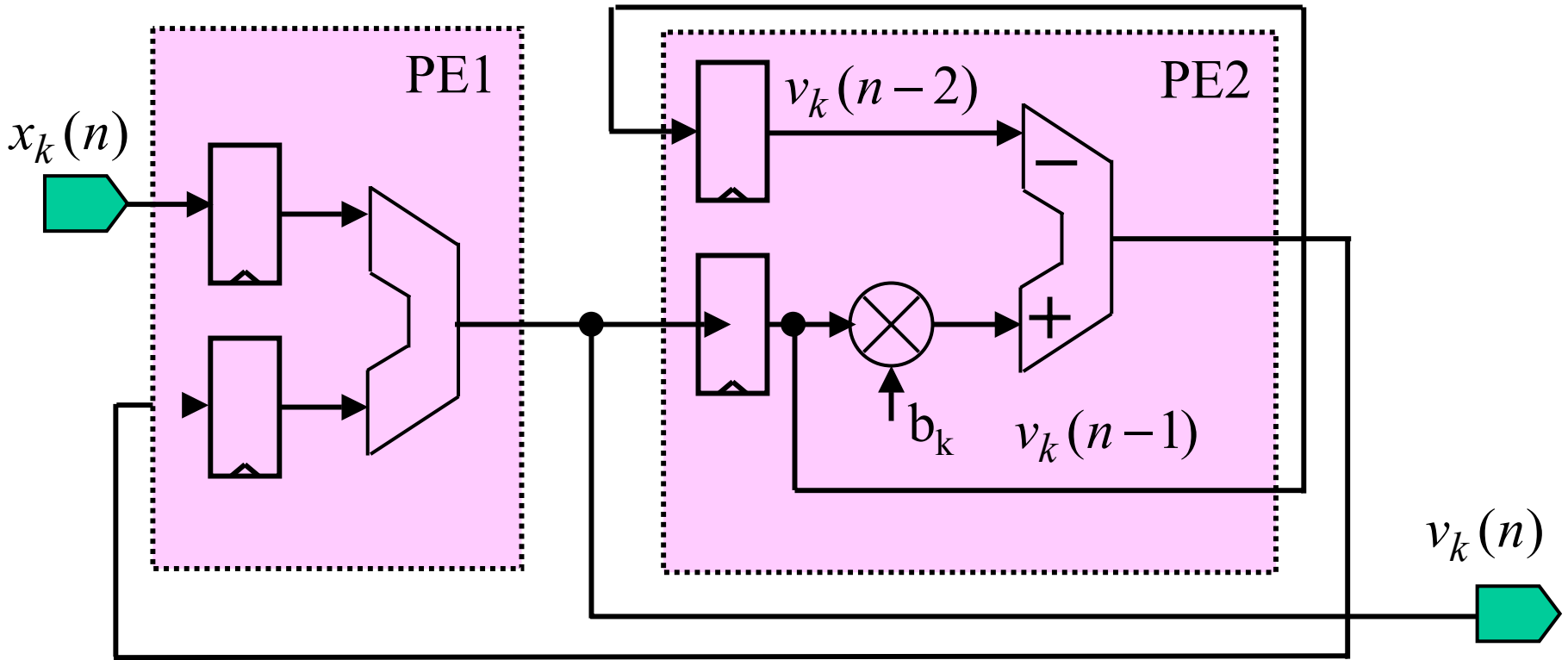
RDPP Multiply Cycles

Assumes RDPP process 8 samples in parallel.



For $N < 256$, Cooley-Tukey Algorithm requires more multiply cycles.

Two PEs Per DFT Value





FTHSI Throughput



Assume RDPP cycle rate = 50 MHz

- $T_{\text{cycle}} = 20 \text{ ns}$
- Not known until hardware design is complete

Compute 8 DFT coefficients in under 2.56 μsec + overhead

- Overhead = configuration and read-out time
- Average 320 nsec/sample + overhead

Option 1: Compute 128-point FFT in 8-sample slices

- Requires 16 passes of input data through RDPP
- Overhead is light: just change b_k values in data registers.
- 41 μsec + overhead for 128 samples

Option 2: Use 16 RDPP chips in parallel.

- Easy to do with Goertzel structure.
- Compute all 128 points in a few microseconds.



Conclusion



The Reconfigurable Data Path Processor exploits data path parallelism for streaming signal processing tasks.

The Cooley-Tukey FFT does not fit the streaming model, and is difficult to parallelize in a data flow processor.

The Goertzel algorithm requires $O(N^2)$ operations, but

1. It is straightforward to implement in a data flow processor.
2. The complex arithmetic operations can be replaced with real-valued operations.
3. For moderately-sized data sequences, the Goertzel algorithm running in the RDPP can offer practical high-speed FFT computation on a parallel processor.
4. We can stack an arbitrary number of RDPP chips to compute the FFT in one pass through the data.