# Design, Implementation and Performance Evaluation of Synthetic Aperture Radar Signal Processor on FPGAs

Hemang Parekh

Masters Thesis
MS(Computer Engineering)
University of Kansas

23rd June, 2000

Committee: Dr. Gary Minden (Chair)
Dr. Joseph Evans
Dr. Arvin Agah

## Presentation Outline

- Motivation
- SAR Signal Processing
- Target Architecture and Design Flow
- Design and Implementation
- Implementation results
- Conclusion
- Future Work

# Some Acronyms used in this presentation

| | |
|---|---|
| ASF | Alaska SAR Facility |
| ACS | Adaptive Computing Systems |
| CCSD | Computer Compatible Signal Data |
| CEOS | Committee on Earth Observation Satellites |
| DARPA | Defense Advanced Research Projects Agency |
| ERS | European Remote Sensing Satellite |
| ESA | European Space Agency |
| FPE | Functional Programming Environment |
| PRF | Pulse Repetition Frequency |
| SAR | Synthetic Aperture Radar |

# Motivation

# Motivation

What's SAR signal Processing?

- Raw data received from SAR
- Ground processing to remove Doppler
- Getting a viewable image

What's so special about it?

- The amount of data is enormous
- The ground processing is computationally intensive

# Motivation

ASIC*, a solution?

- ASIC : Optimized hardware solution
  - But, large time to market
  - Not flexible

Why FPGAs ?

- FPGA : Field Programmable Gate Arrays
  - Optimized hardware solution
  - Also, small time to market
  - Flexibility equivalent to a software implementation

*Application Specific Integrated Circuit

# SAR Signal Processor

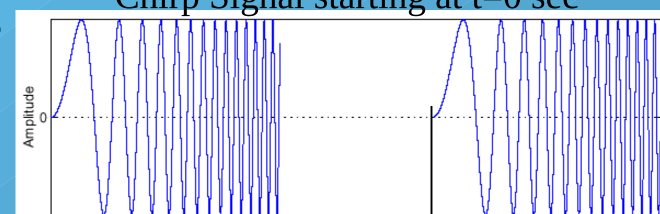
# SAR Signal Processing

- Synthetic Aperture Radar
  - Improved resolution compared with conventional radar
  - Chirp improves the range resolution

$$d_R = \frac{c t_p}{2}$$ … single freq. pulse

$$d_R = \frac{c}{2B}$$ … chirp pulse

Chirp Signal starting at t=0 sec



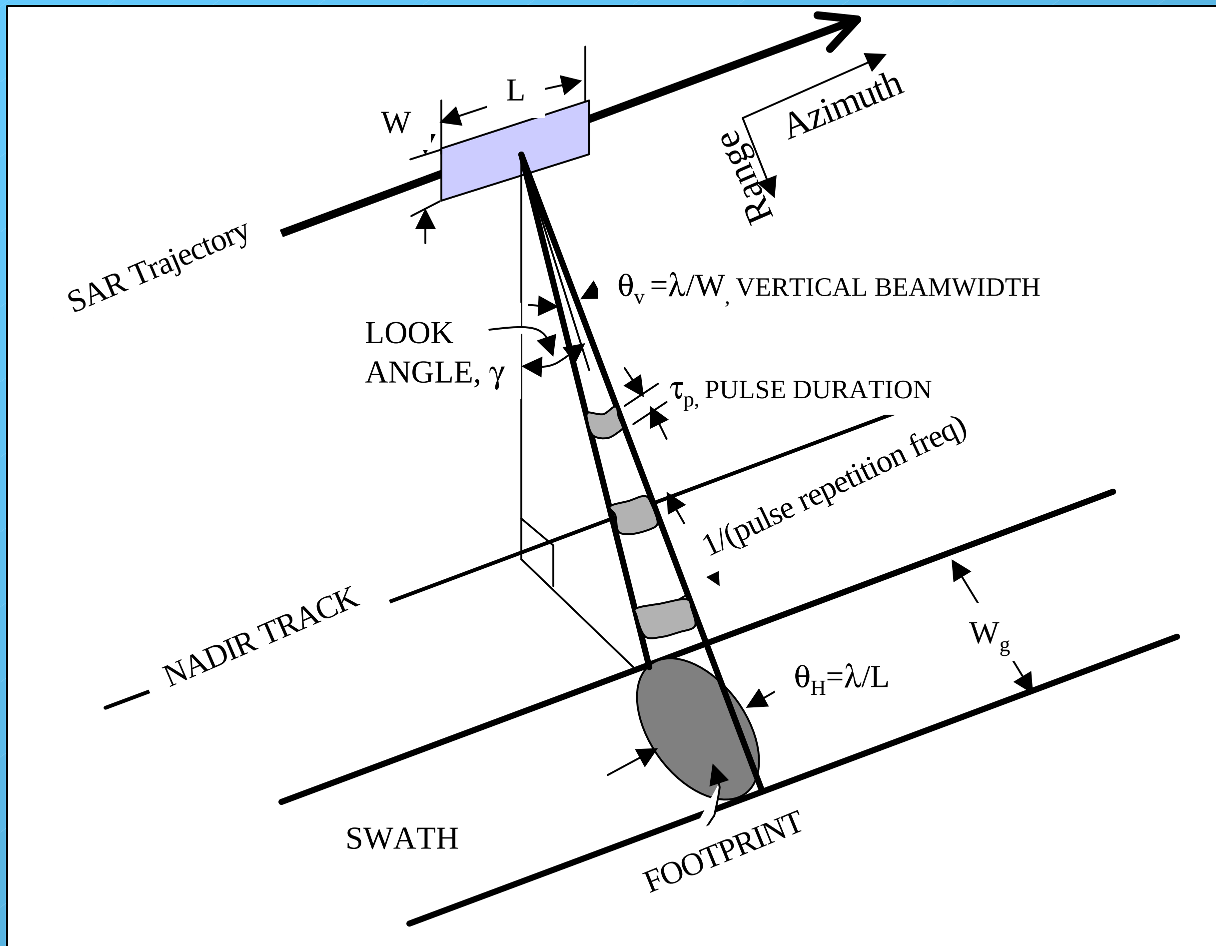

t=0 $\tau_p$ 1/PRF

- Doppler improves the azimuth resolution

$$d_A = \frac{R l}{A}$$ … R:range, A: antenna size, λ:wavelength
- limited by antenna size A

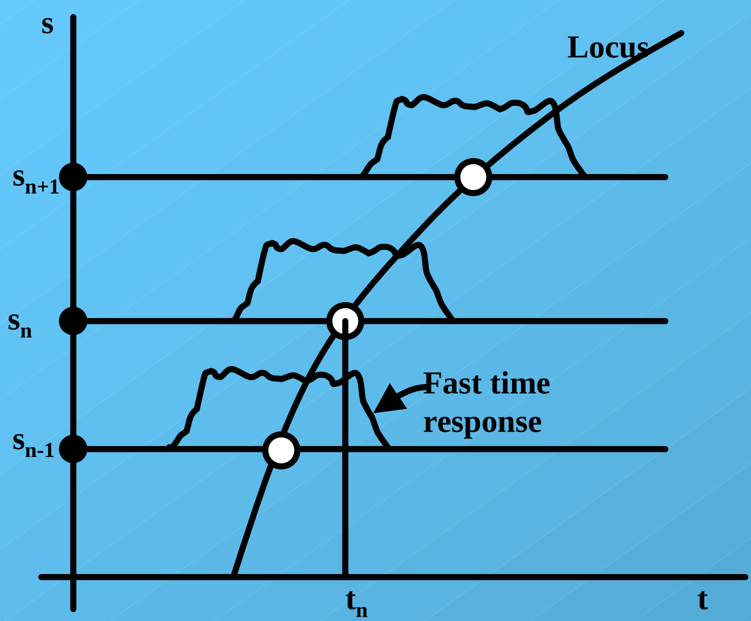
$$d_A = \frac{L}{2}$$ … L:antenna aperture length

# Synthetic Aperture Radar

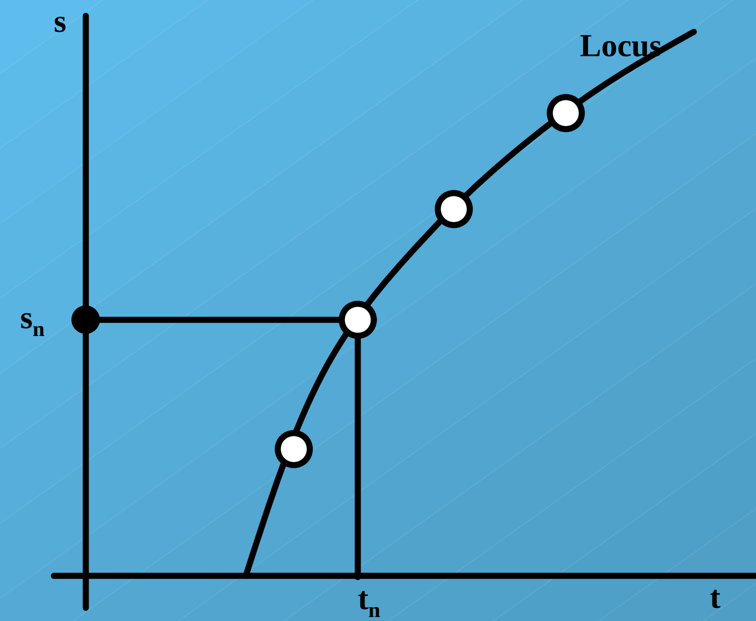



## SAR scan geometry

# SAR Signal Processing




Response of echo from point target




Point Target Response after "collecting" information from Fast time response

# SAR Signal Processing

Some Facts and Specs.

- Software Signal Processor, *aisp*, from ASF
- Raw Data from ERS-1 satellite, in 5 bit real and 5 bit imaginary, zero-padded to make up a byte
- A patch of 4096 x 4096 data pixels considered.

# SAR Signal Processing

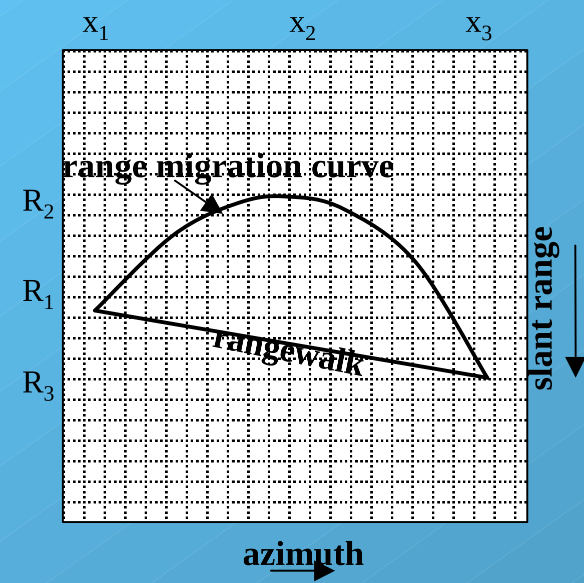
## Range Processing

- Complex raw data correlated with a matched filter (same for all azimuth distance)
- Matched filter response obtained from the chirp characteristics

## Azimuth Processing

- Range Compressed data correlated with a matched filter
- Matched filter response obtained from the Doppler history of the echoed signal
- Matched filter different for each range line.

# Range Migration

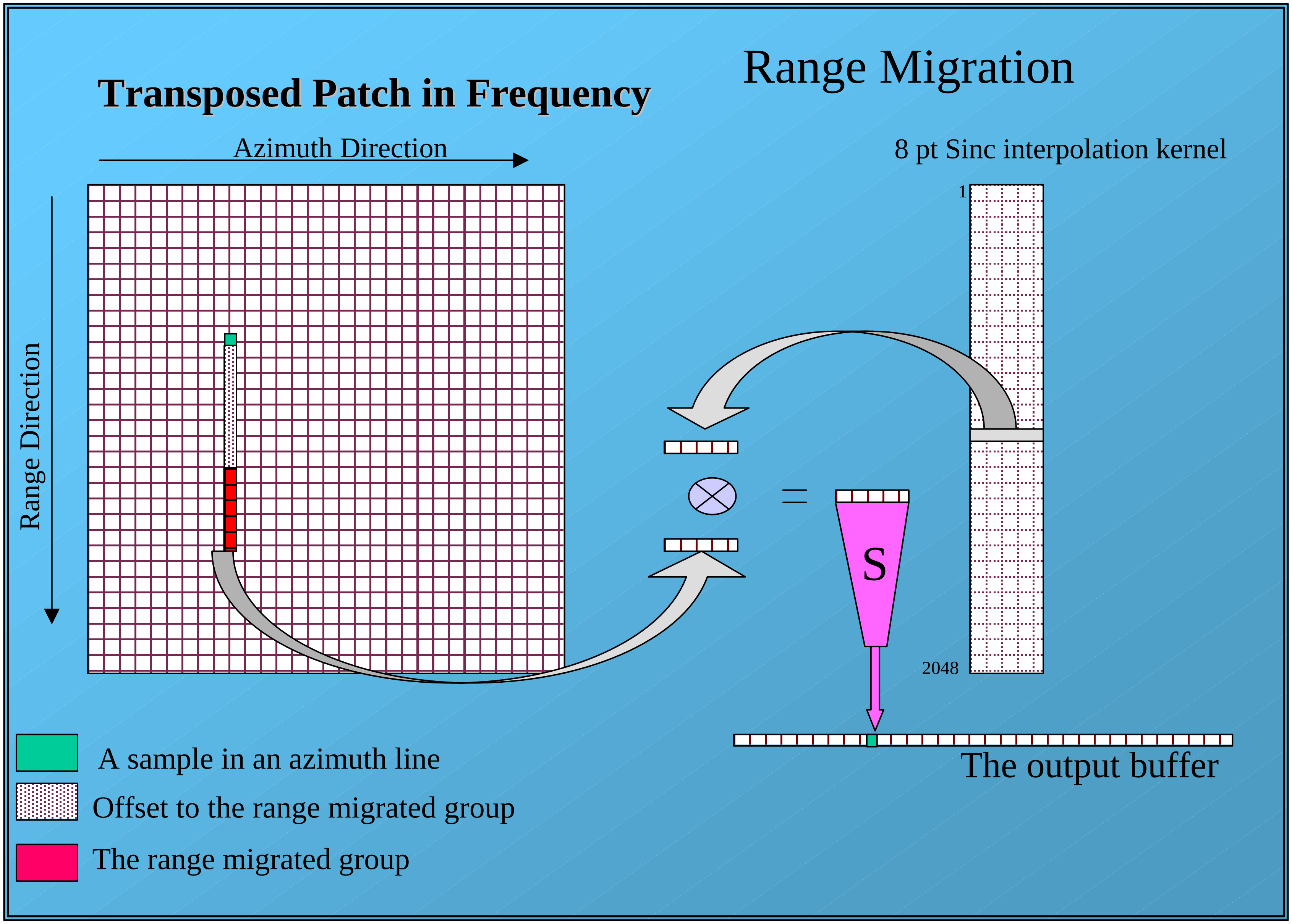



- After Range Compression
  - Input from one range line can appear at output of matched filter,delayed and associated with next range line.
- For SAR, this Doppler induced error inevitable.
- Doppler-induced shift - range curvature
- earth rotation causes - range walk
- resultant migration path - parabolic

- However, for ERS-1,calculating using orbit parameters, the maximum migration, a couple of range lines.
- This feature exploited in implementation.

- Since Doppler induced migration,
  range migration correction in Doppler freq. domain

- Since the calculated value of migration will not be a whole integer,
  interpolation is used. ASF uses an 8 point interpolation vector for range migration correction.



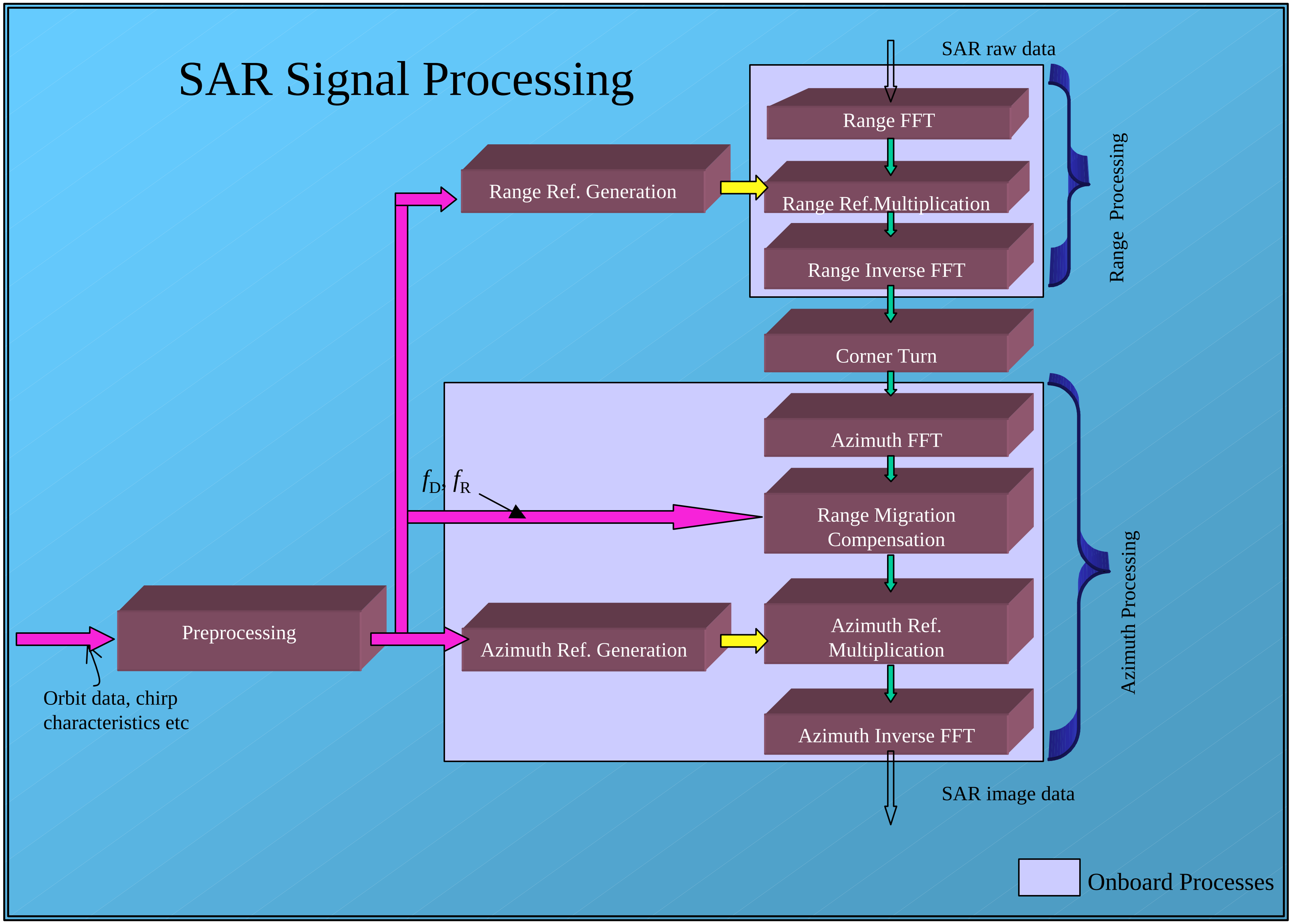
Range Migration
Transposed Patch in Frequency
Azimuth Direction
Range Direction
8 pt Sinc interpolation kernel
1
2048
S
The output buffer
A sample in an azimuth line
Offset to the range migrated group
The range migrated group

# SAR Signal Processing

- Main units of Signal Processing are:
  - Matched Filtering for range compression
  - Matched Filtering for azimuth compression
  - Range Migration Correction

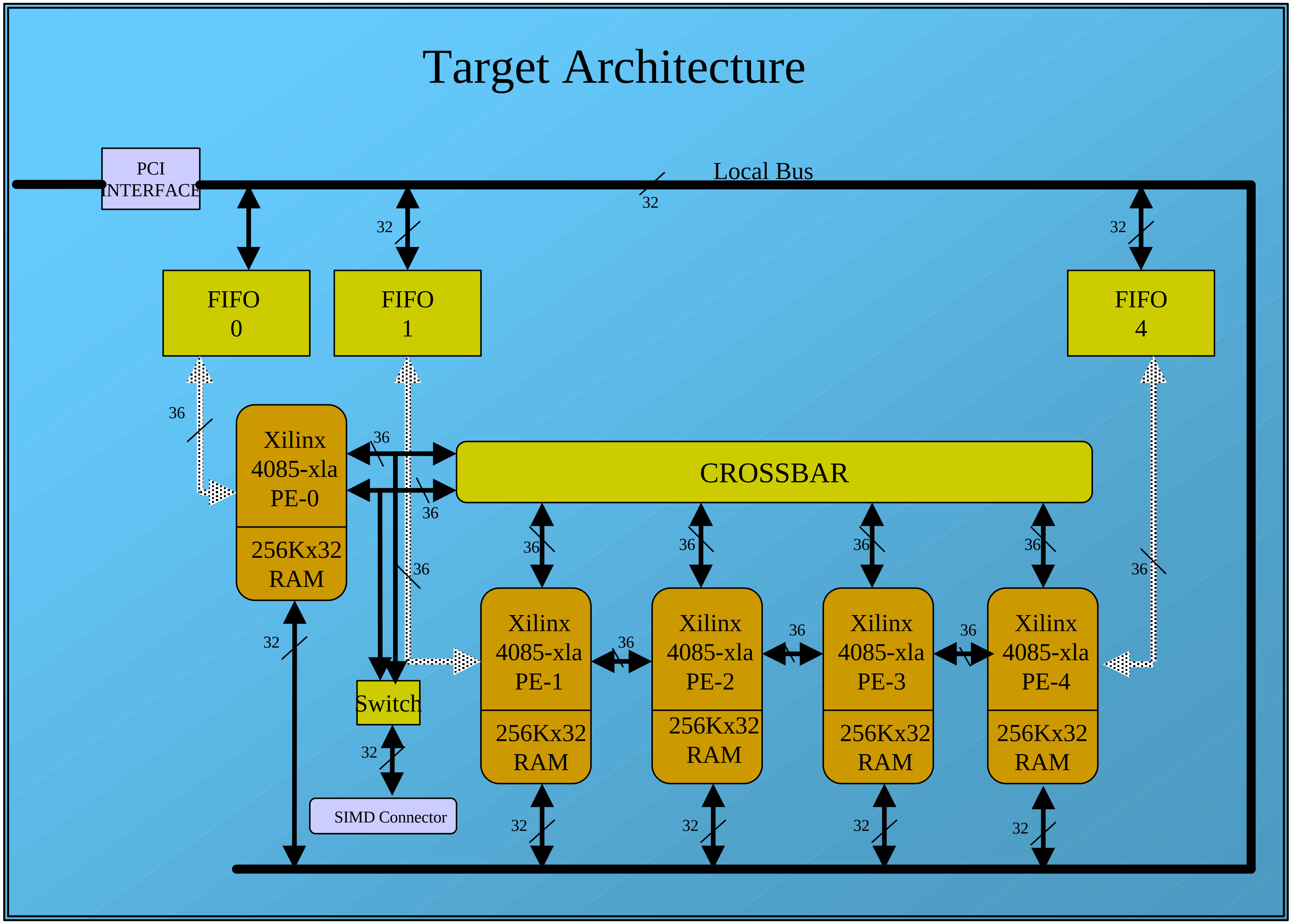
- Correlation works faster by
  - Taking Fourier transform of input (time--> frequency)
  - Multiply it with complex conjugate of frequency response of matched filter
  - Taking inverse Fourier transform (frequency --> time)

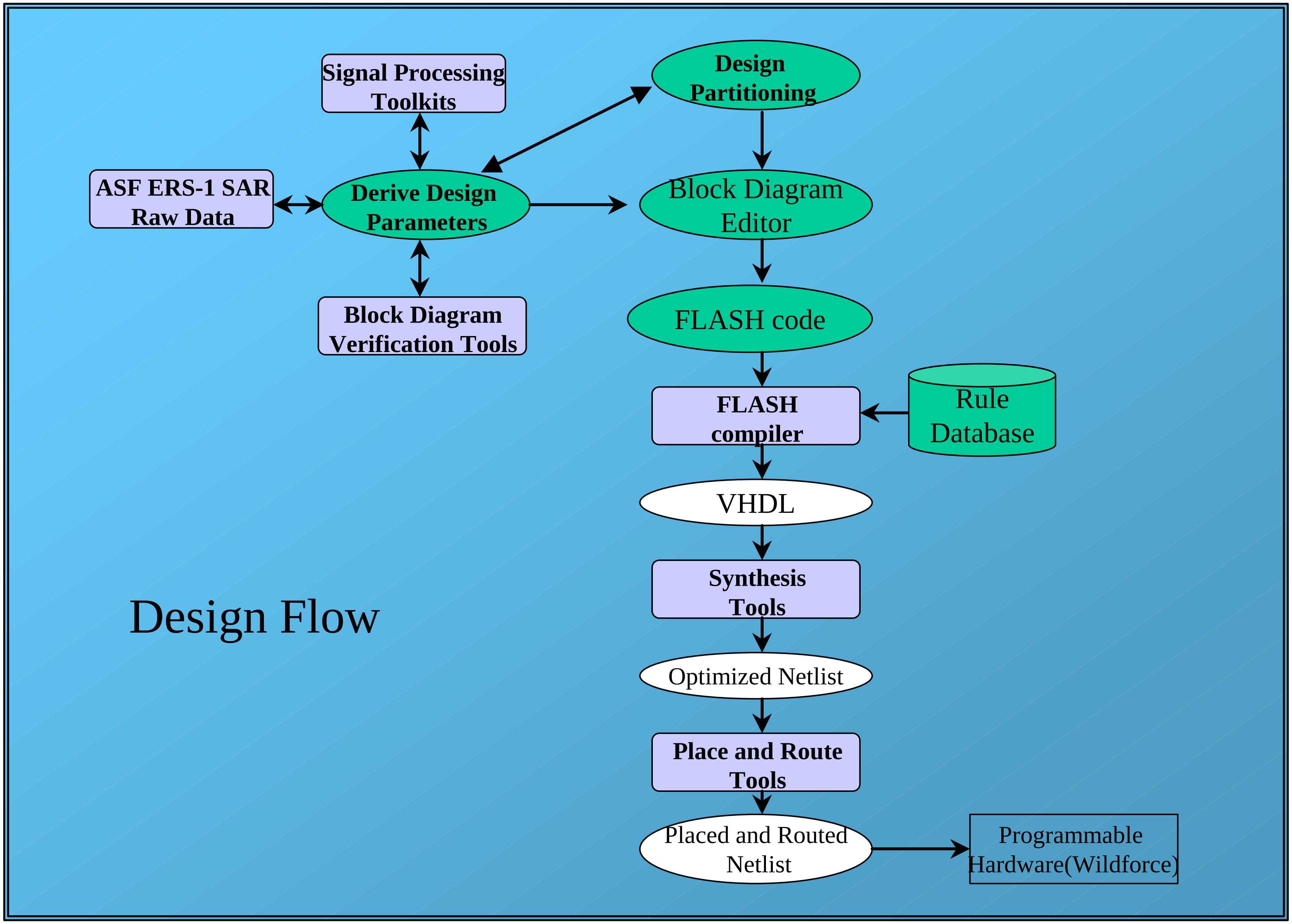
- Hence an efficient FFT algorithm(4096 point FFT) would be required.



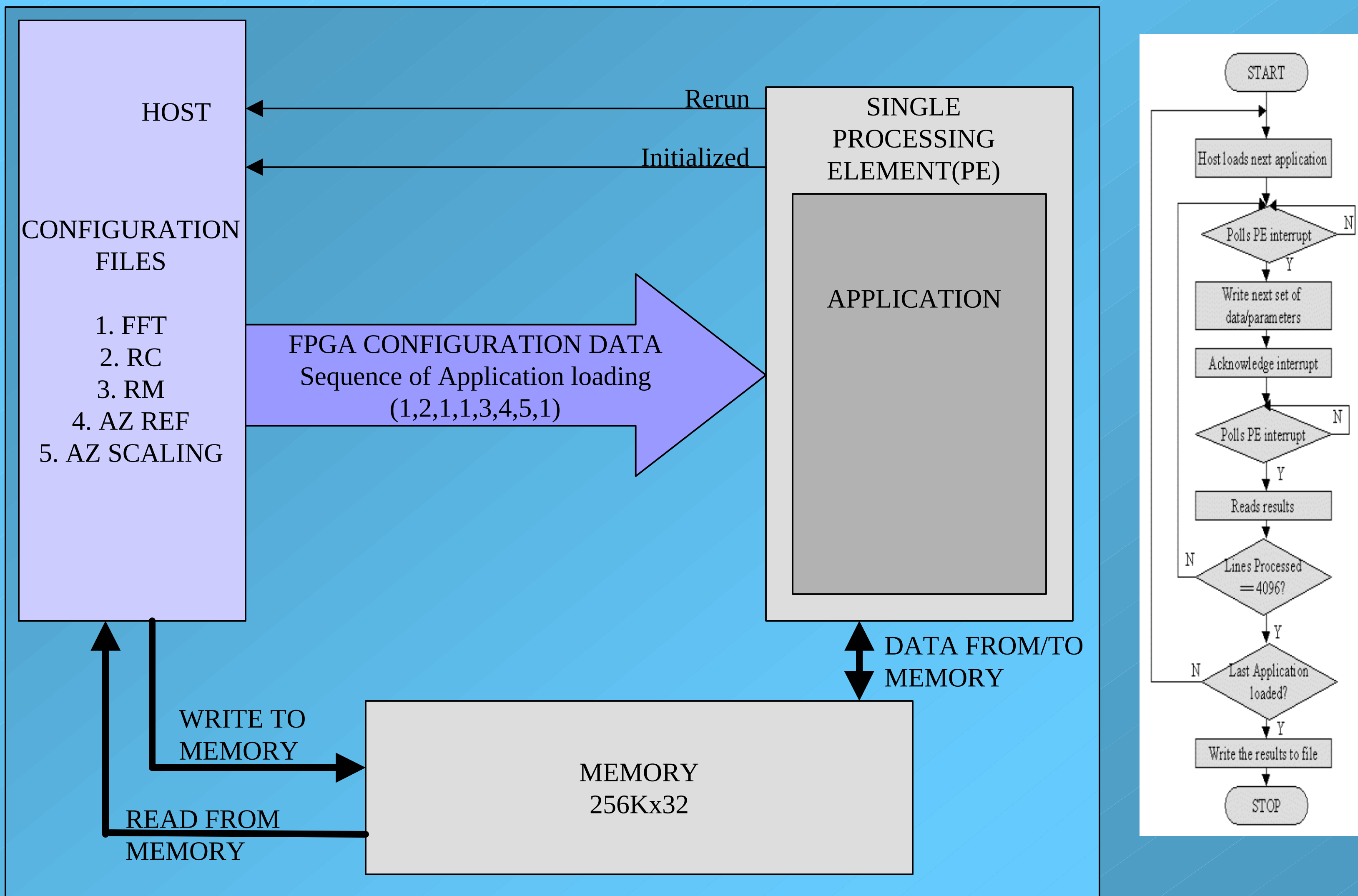
SAR Signal Processing
SAR raw data
Range FFT
Range Ref. Generation
Range Ref.Multiplication
Range Inverse FFT
Range Processing
Corner Turn
Azimuth FFT
$f_D$, $f_R$
Range Migration Compensation
Preprocessing
Azimuth Ref. Generation
Azimuth Ref. Multiplication
Azimuth Inverse FFT
Azimuth Processing
Orbit data, chirp characteristics etc
SAR image data
Onboard Processes

# Target Architecture and Design Flow


Target Architecture
PCI INTERFACE
Local Bus
32
32
32
FIFO 0
FIFO 1
FIFO 4
36
36
36
36
Xilinx 4085-xla PE-0
256Kx32 RAM
CROSSBAR
36
36
36
36
36
Xilinx 4085-xla PE-1
256Kx32 RAM
Xilinx 4085-xla PE-2
256Kx32 RAM
Xilinx 4085-xla PE-3
256Kx32 RAM
Xilinx 4085-xla PE-4
256Kx32 RAM
36
36
36
32
Switch
32
SIMD Connector
32
32
32
32


Signal Processing Toolkits
Design Partitioning
ASF ERS-1 SAR Raw Data
Derive Design Parameters
Block Diagram Editor
Block Diagram Verification Tools
FLASH code
FLASH compiler
Rule Database
VHDL
Synthesis Tools
Design Flow
Optimized Netlist
Place and Route Tools
Placed and Routed Netlist
Programmable Hardware(Wildforce)

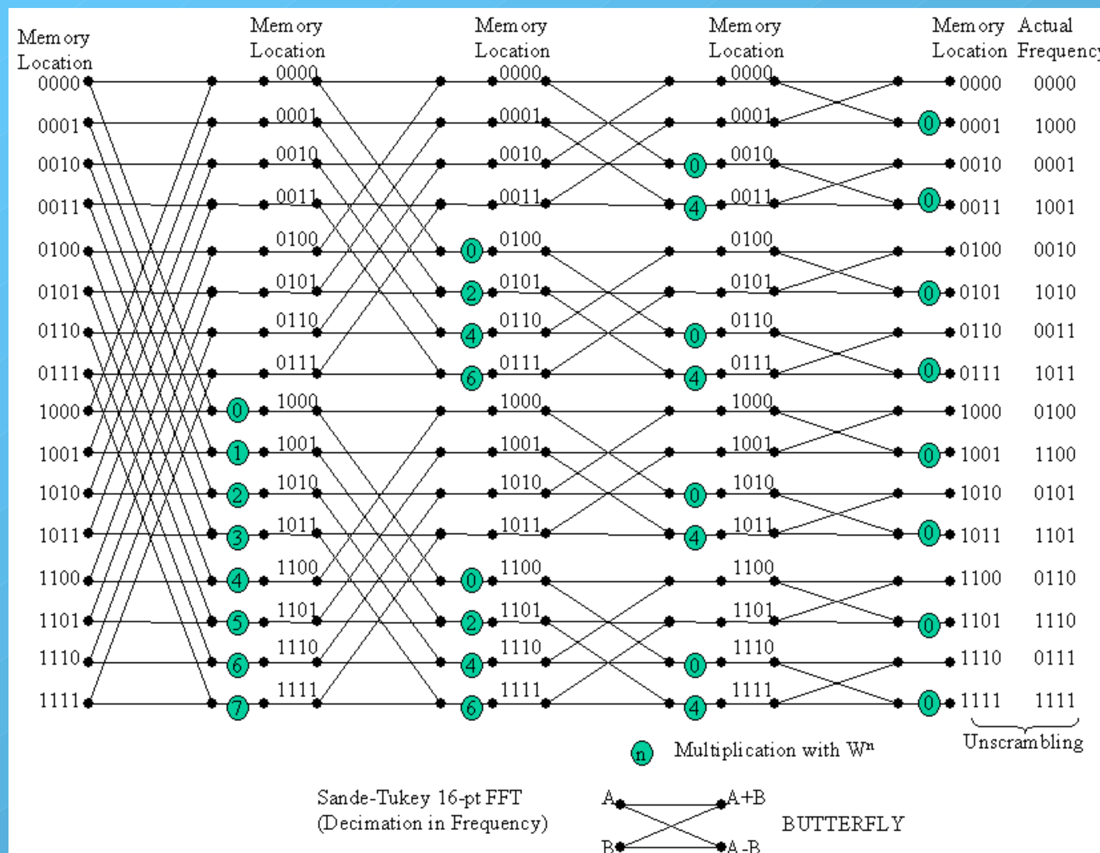
# HOST-PE Interaction Model



# Design And Implementation Of SAR Signal Processor On FPGAs

# Fast Fourier Transform

- Sande-Tukey Algorithm - Decimation in Frequency (Radix -2)



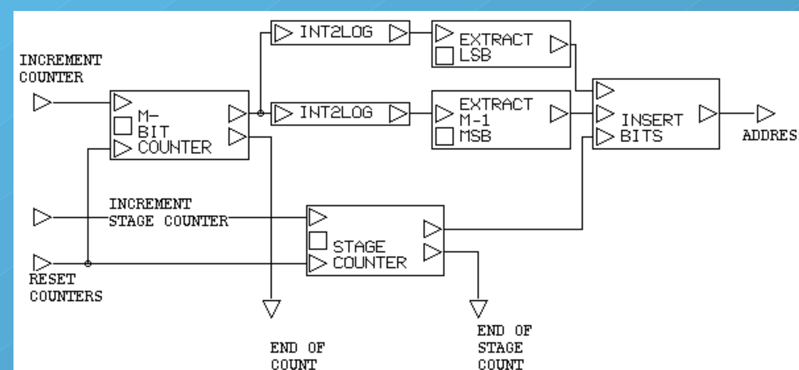

16 point FFT

# Fast Fourier Transform

- Address Generation

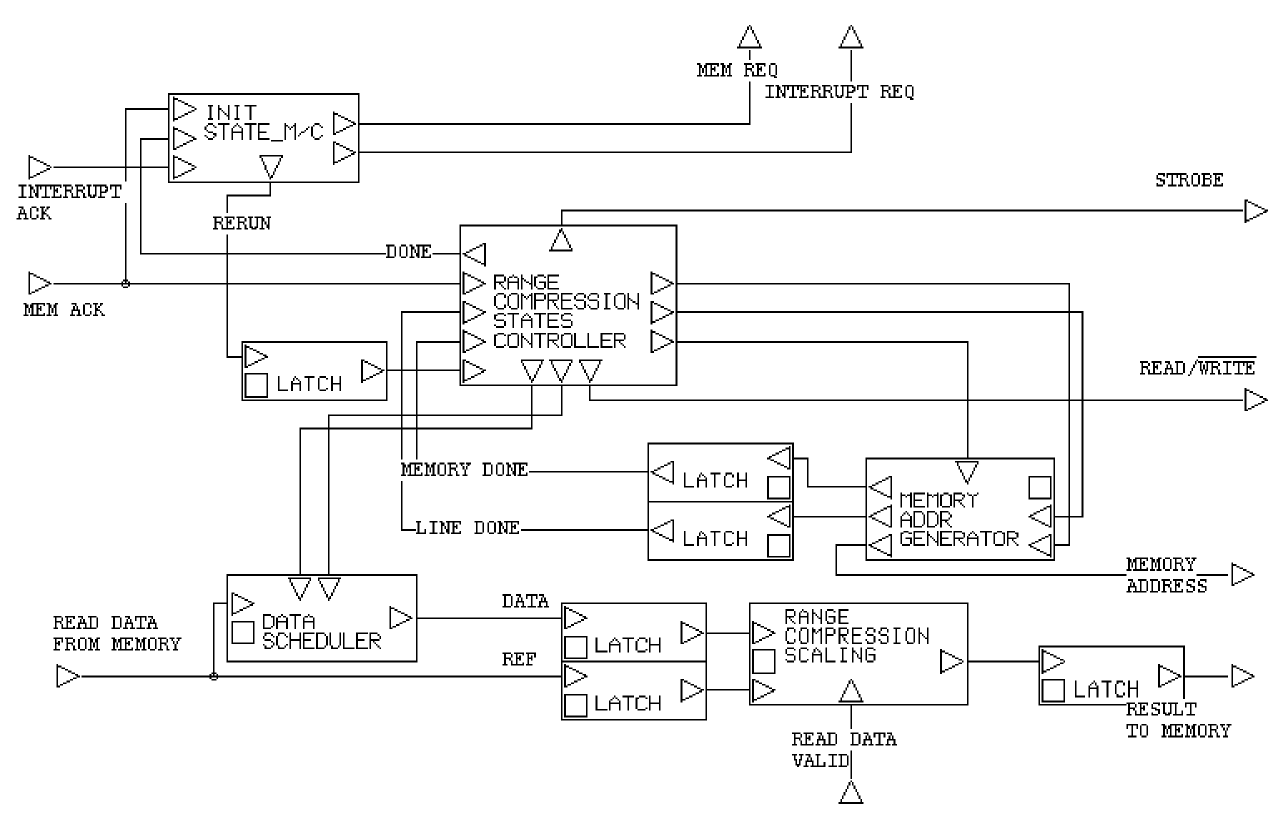
| Stage 0 | Stage 1 | Stage 2 | Stage 3 |
|---|---|---|---|
| 000 0 | 0000 | 0000 | 0000 |
| 000 1 | 0100 | 0010 | 0001 |
| 001 0 | 0001 | 0001 | 0010 |
| 001 1 | 0101 | 0011 | 0011 |
| 010 0 | 0010 | 0100 | 0100 |
| 010 1 | 0110 | 0110 | 0101 |
| 011 0 | 0011 | 0101 | 0110 |
| 011 1 | 0111 | 0111 | 0111 |
| 100 0 | 1000 | 1000 | 1000 |
| 100 1 | 1100 | 1010 | 1001 |
| 101 0 | 1001 | 1001 | 1010 |
| 101 1 | 1101 | 1011 | 1011 |
| 110 0 | 1010 | 1100 | 1100 |
| 110 1 | 1110 | 1110 | 1101 |
| 111 0 | 1011 | 1101 | 1110 |
| 111 1 | 1111 | 1111 | 1111 |

- Inserting the LSB at correct location
- Correct location: (bitwidth - stage_num.)



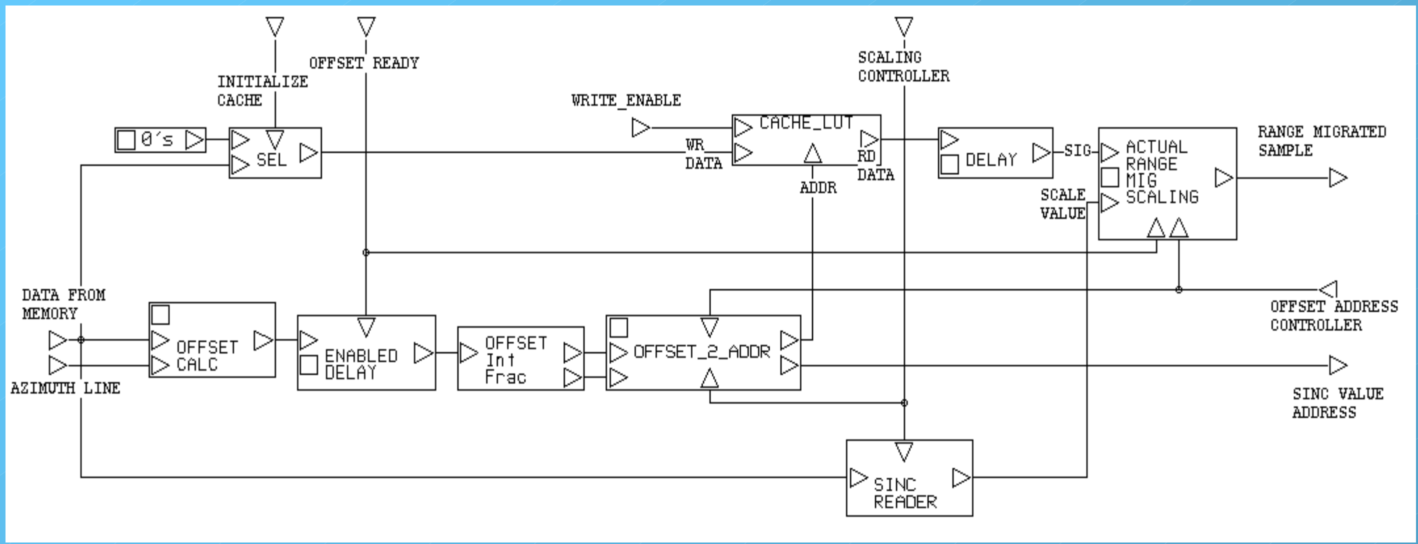

BDE Diagram for FFT Address

# Range Scaling




BDE Diagram for RC

# Range Migration

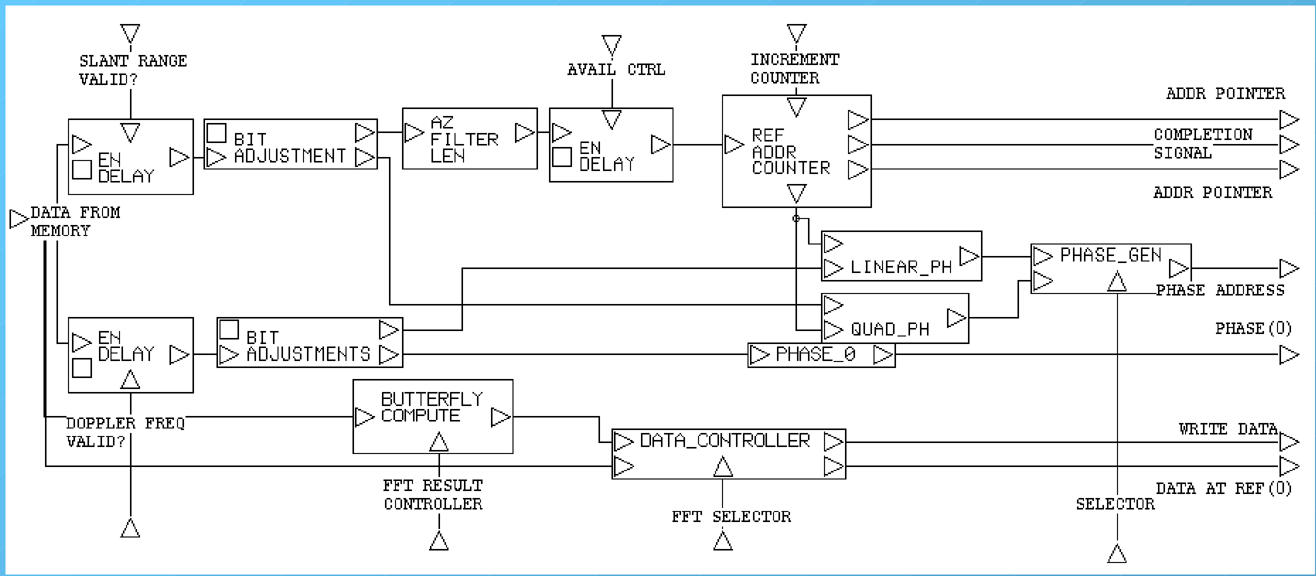



BDE Diagram for RM

Cache

- Since offset Maximum 3-4 range lines
- 8-point interpolation
- so data along range required is ~12 and so
- CACHE of 16 next possible values

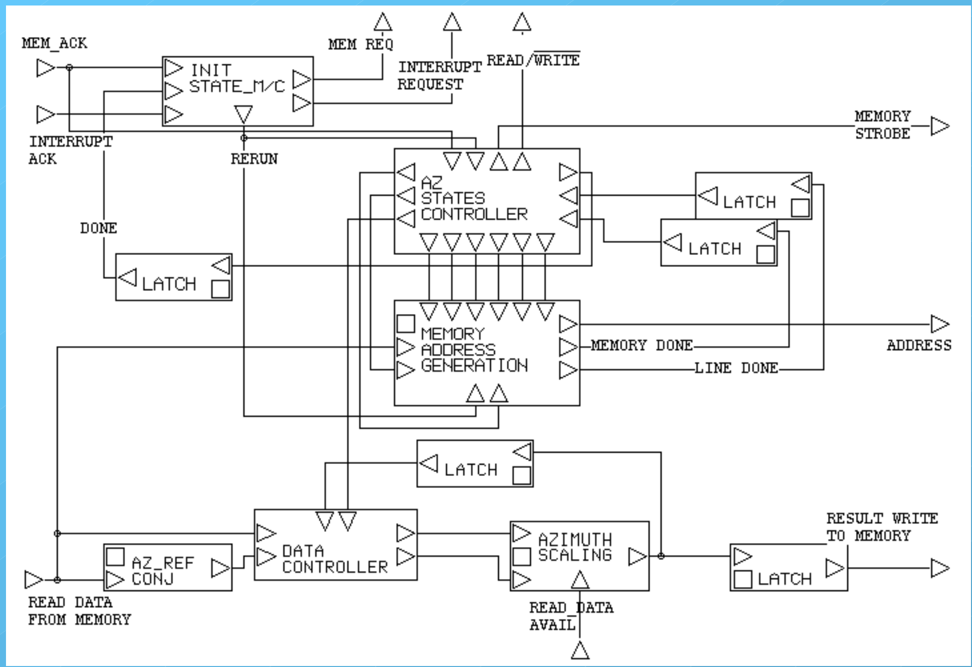
# Azimuth Reference Generation




BDE Diagram for AZREF

- •Calculated from precomputed Doppler history
  - • Doppler history -- Doppler Centroid, and Doppler rate

# Azimuth Scaling




BDE Diagram for AC

- Similar to Range Scaling
    - Except that the reference is unique for each azimuth line

# Implementation Figures

| Modules | FFT | RC | RM | AZ_REF | AZ_C |
|---|---|---|---|---|---|
| **% Device Util.** | 40 | 37 | 61 | 63 | 50 |
| **Cycles/memory** | 10322000 | 921700 | 4671200 | 10679000 | 1843300 |
| **Clock (MHz)** | ~13 | ~14 | ~10 | ~10 | ~13 |
| **Time (sec)/memory** | 0.794 | 0.066 | 0.467 | 1.068 | 0.142 |
| **Mem. reload/patch** | 60 | 60 | 60 | 60 | 30 |
| **Time (sec)/patch*** | 11.12 | 0.924 | 6.538 | 14.95 | 3.976 |

Note: Lines/patch = 4096

**Time to process a module**

- FFT most time intensive of all the processes
- 4 FFT modules per patch. (R.FFT, R.IFFT, A.FFT, A.IFFT)
- Total time: 4(11.12) + 0.9 + 6.5 + 14.95 +3.98
  = 70.81 seconds

*For #PEs = 5

# Results
# and
# Conclusion

# Results

Comparing Images

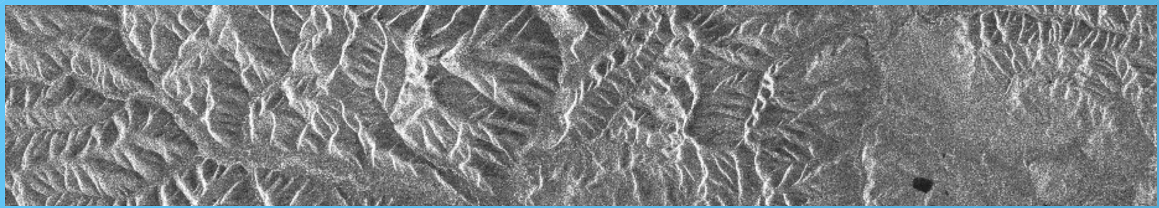


Image from ASF Software

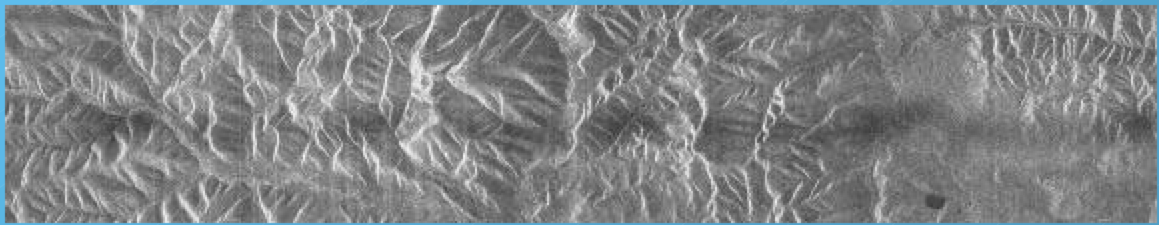
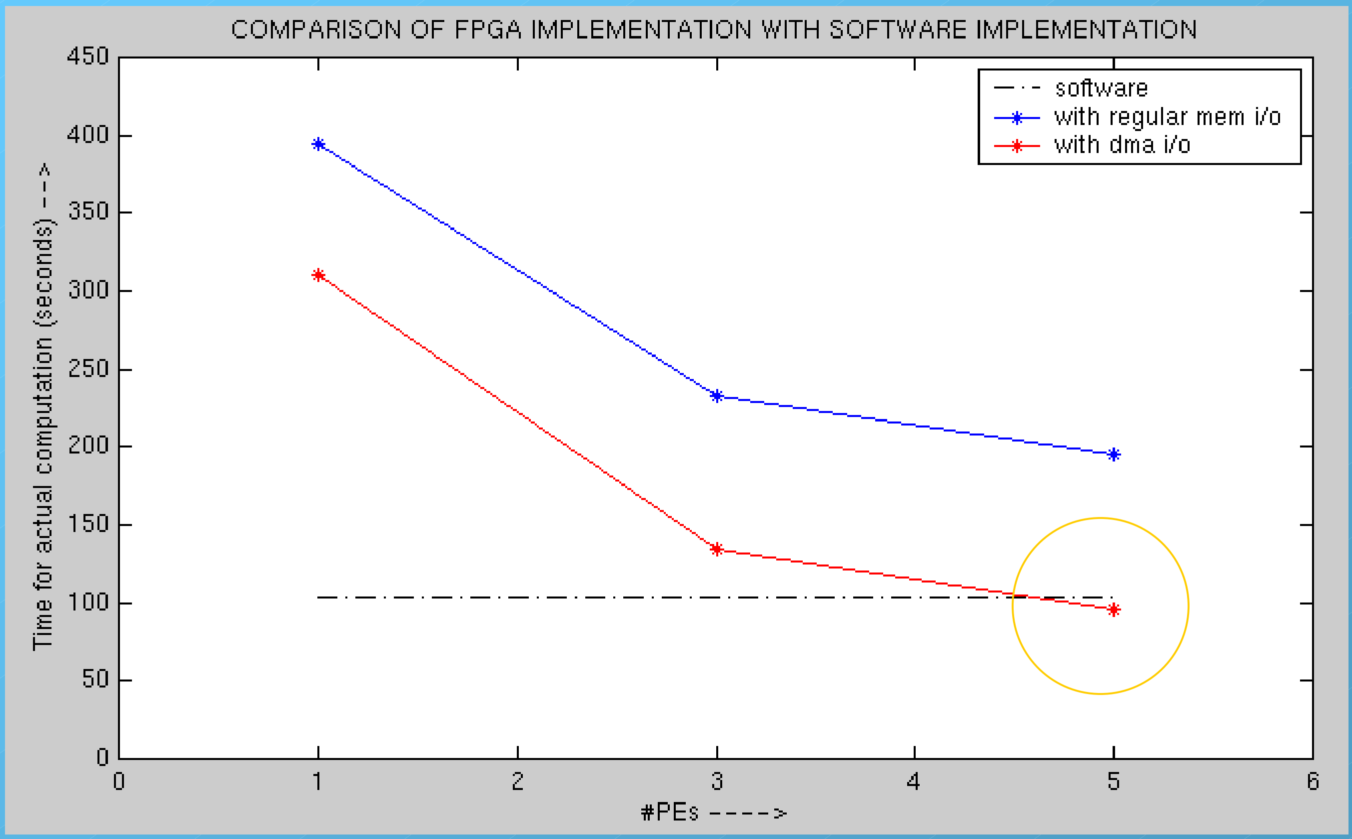


Image from FPGA implementation

# Comparison with Software Implementation

| Implementation | | Time for actual computation (sec)/ Total time (sec) |
|---|---|---|
| **Software - ASF's aisp utility** | | 103/106.4 |
| **Using regular memory transfer** | | |
| #PEs | 1 | 394.9/402.34 |
| | 3 | 232.43/240.05 |
| | 5 | 195.69/203.32 |
| **Using DMA memory transfer** | | |
| #PEs | 1 | 310.57/318.13 |
| | 3 | 136.37/145.6 |
| | 5 | 95.32/102.56 |

# Comparison with Software Implementation



# Conclusion

- Demonstrates the design and FPGA implementation of certain SAR algorithms.
- Shows efficient use of reprogrammability of FPGA by loading different stages of the signal processor.
- This work also shows the effectiveness of FLASH and BDE in implementing such a complex design.
- SAR image generated shows a successful implementation of a prototype SAR signal processor on FPGA.

- However, memory I/O, a bottleneck. larger memory size should provide better performance.

# Future Work

Some of the possibilities...

- Incorporate speckle reduction, Interferometry etc
- Higher Radix FFT usage
- Incorporate preprocessing on FPGAs.
- Generalize design for other Radar like SIR-C, JERS, etc.
- Floating point implementation to increase resolution

Questions?