Digital Generation and Phasing of Transmitter Signals for SuperDARN Class Radars

By J.S. Whittington, J.C. Devlin, and T. Salim.

La Trobe University

Department of Electronic Engineering,
La Trobe University

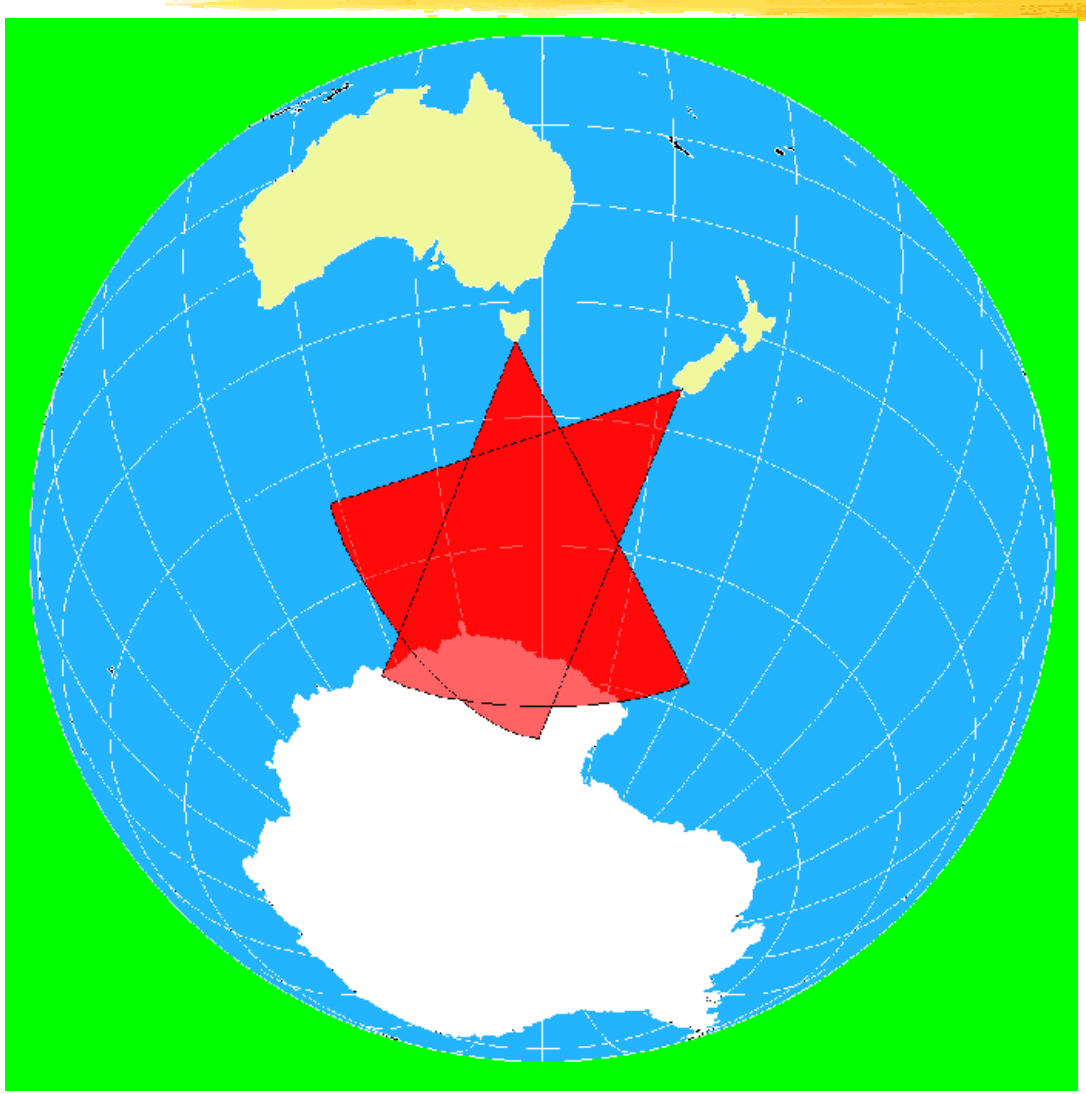
Talk Overview

- Digital Radio and Digital TIGER
- Proposed Digital Transmitter System
- Digital Frequency Generator
  - Design and implementation
- Phasing Matrix
  - Poly-phase filter, simulation and implementation
- Discussion and Conclusion

SuperDARN Radars

- Super Dual Auroral Radar Network
  - 15 separate radars in higher latitudes of the northern and southern hemispheres – more planed
  - An over-the-horizon radar technique
  - Used to measure plasma motion in the Earth's upper atmosphere
- Each radar uses 16 main antennas
  - Transmits 16 beams of 3 - 4 degrees each, over a 52 degree arc
  - Transmission frequencies 8MHz to 20MHz

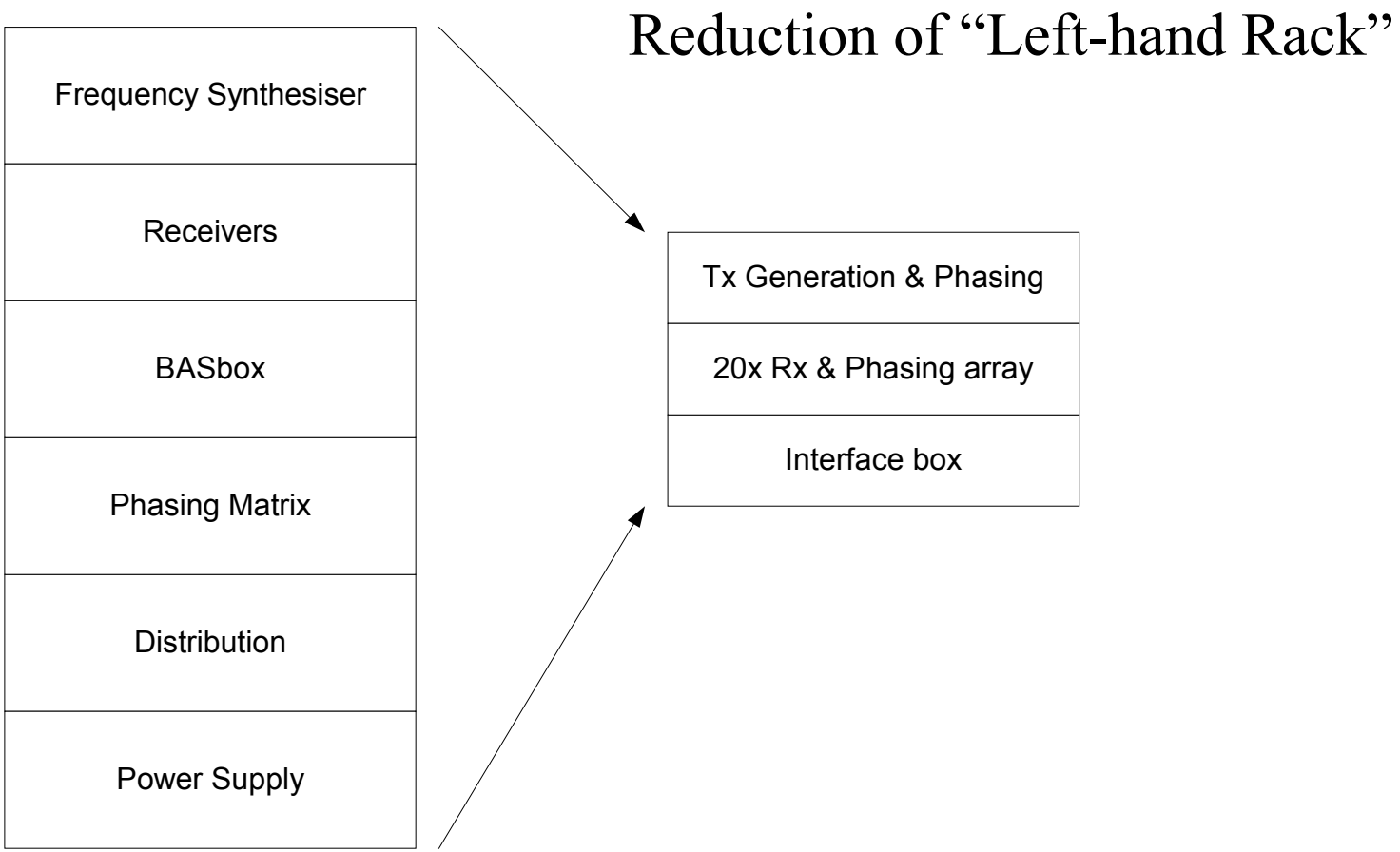
The TIGER Radar



The TIGER Radar

TIGER N.Z. Radar

- a digital evolution of the current analog system



Digital Radio

- Digital signal processing has become an integral part of wireless communications systems
  - Increasingly, replace analog with DSP
  - But also, can employ DSP techniques in ways that have no direct analog counter part
  - DSP solutions well placed to take advantage of advances in Integrated Circuit technology
  - Moore's Law: every 18 months
    - Gate count doubles
    - Frequency increases 50%

Digital Radio

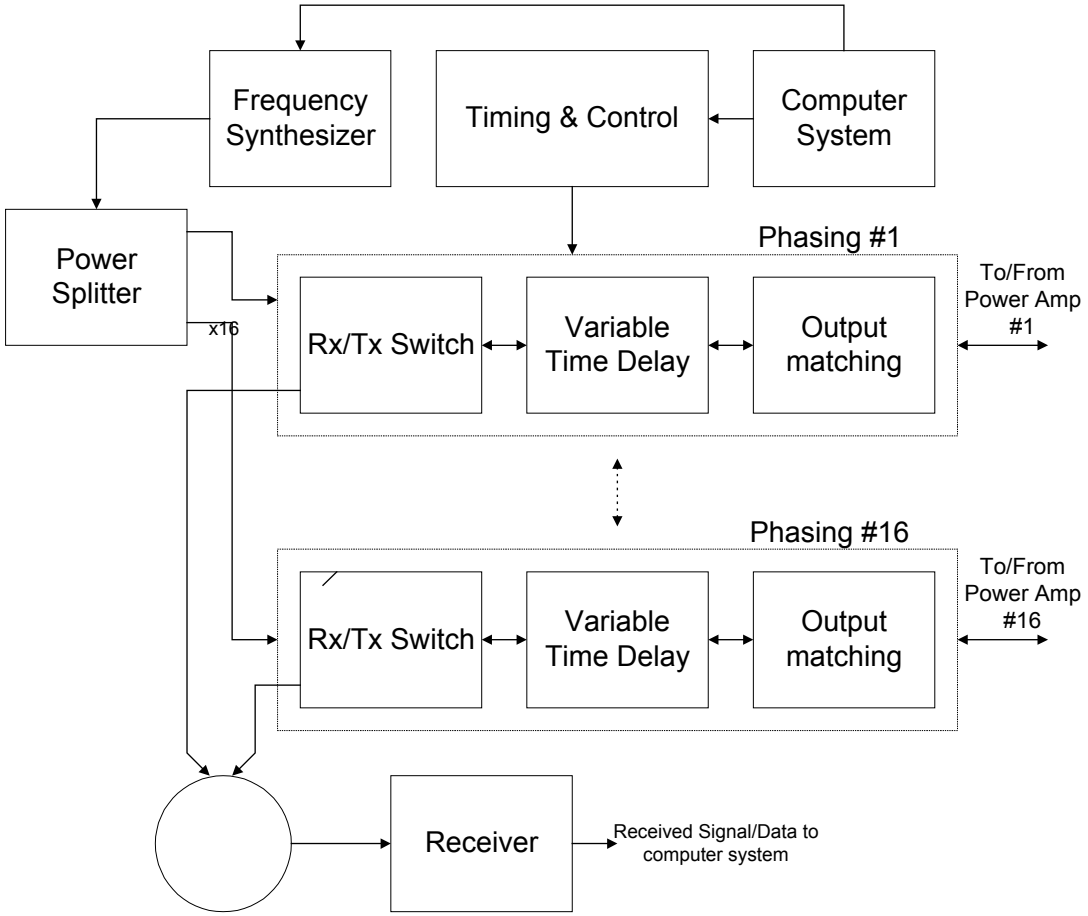
- Example: Mobile Phones
  - Core functions now entirely digital
  - Extensive use of DSP
  - ASIC implementation
    - Good for high speed applications
    - Good for high volume
      - Needed to offset high IC fabrication set up cost

Digital TIGER

- FPGA Implementation. Why?
  - Cost! – low volume
  - Fast design turn around
  - Reconfigurable hardware
    - Different scan modes using same hardware

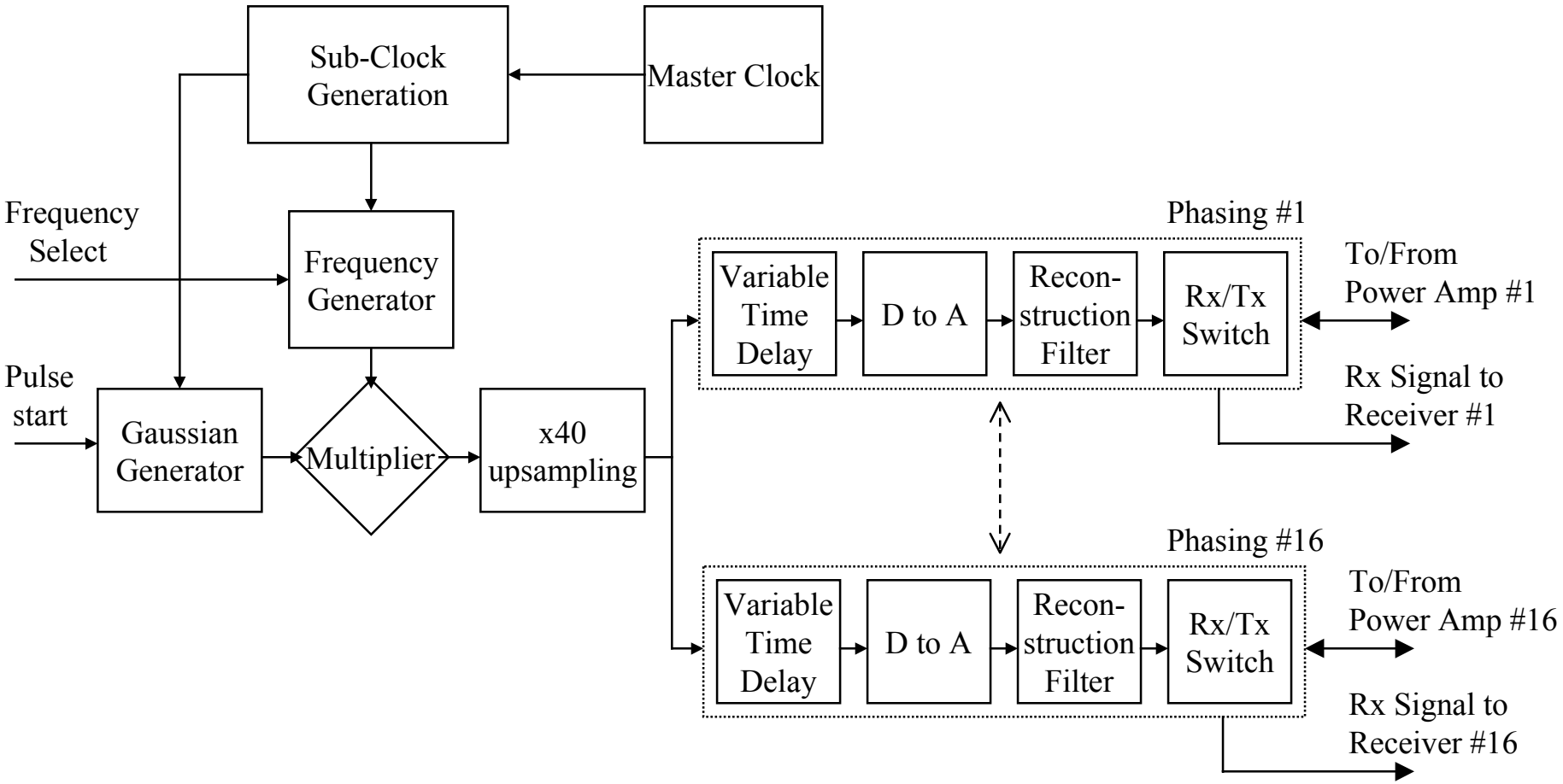
      => just load new configuration
- FPGA performance roughly an order of magnitude behind ASIC
  - Presently, can only produce sections in reasonable cost FPGAs
  - However, Moore's law also applies to FPGAs
  - Plus, vendors now producing FPGAs tailored to DSP
    - e.g. Xilinx: Virtex-II contains embedded 18-bit multiplier resources

Current Analog System



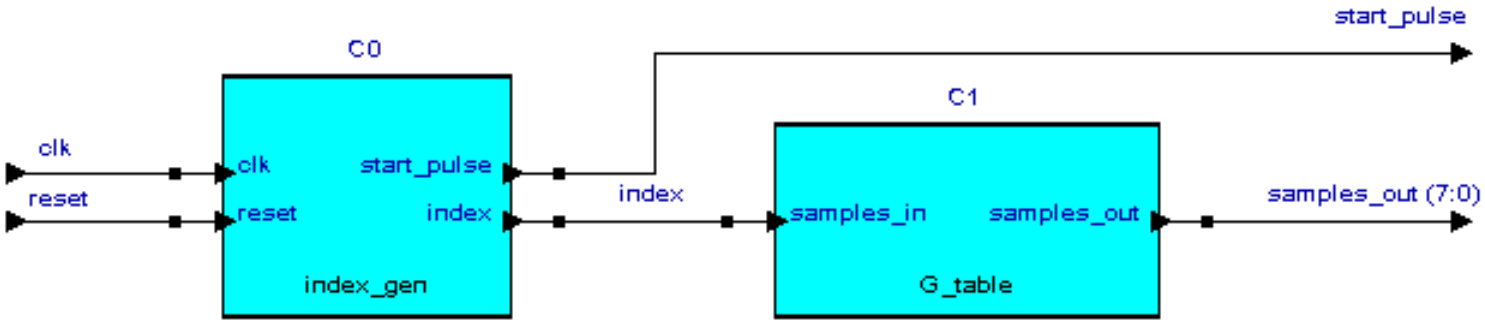
SuperDARN Radar Simplified Block Diagram

Proposed Digital Transmitter System

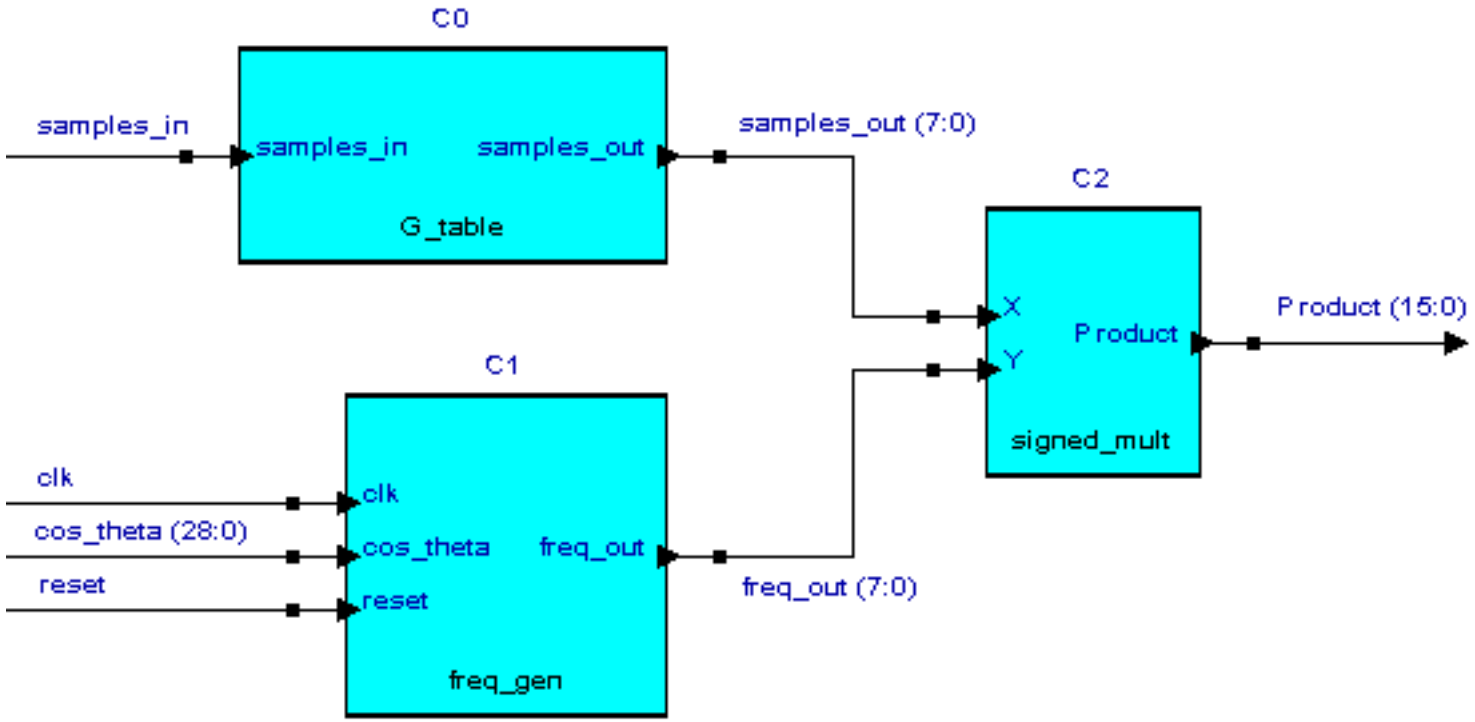


Proposed Block Diagram for Digital SuperDARN Radar

Gaussian Pulse Generator



Multiplier


C0
samples_in
samples_in
samples_out
G_table
samples_out (7:0)
C2
X
Y
Product
signed_mult
Product (15:0)
C1
clk
cos_theta (28:0)
reset
clk
cos_theta
reset
freq_out
freq_gen
freq_out (7:0)

Phasing Matrix

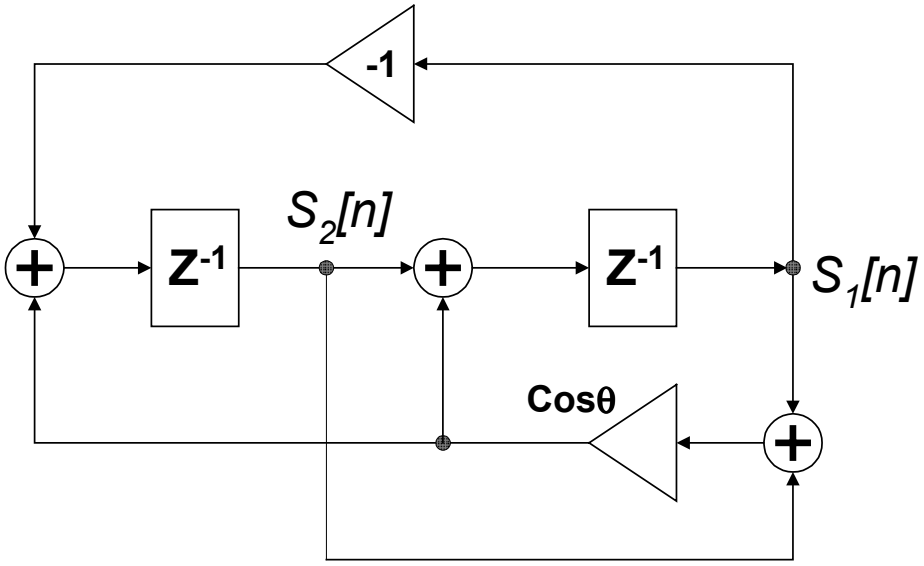
- Could be implemented as a series of time delays
  - Would require roughly a 0.1ns clock (~10Ghz) and a large number of delay registers
  - Expensive in real estate, plus 10GHz clock rate not possible even in present ASIC technology
- More realistic implementation: poly-phase filter
  - This time-variant technique allows is phasing and upsampling to be done in one step
  - Thus, guassian generator, multiplier and frequency generator can operate at a much lower frequency

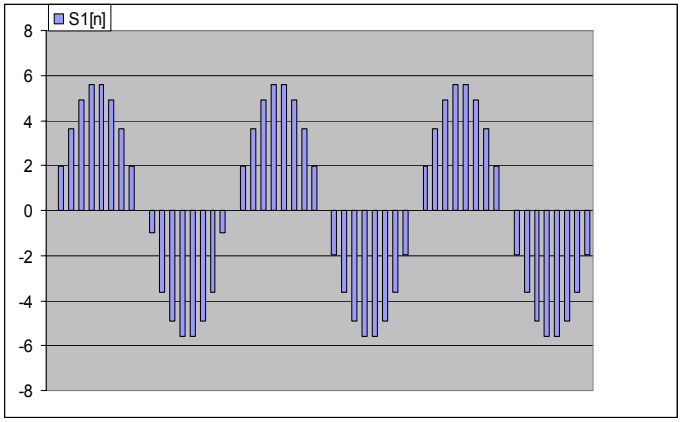
DtoA and Reconstruction Filter

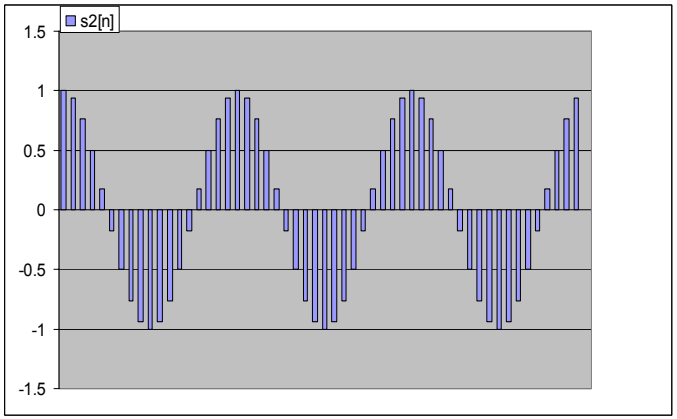
- DtoA quantization noise floor - minimum of 80dB
  - Analog system - 72dB (worst case) with spurii and noise, also harmonics at only 36dB (worst case)
- Reconstruction filter (Analog)
  - A 5 pole filter provides 100dB/decade roll-off
  - Set the sample rate to be 10 times the niquest frequency (40MHz) of the max. signal (20MHz) - gives a sample rate of 400MHz

Frequency Generator

- Based on single multiplier sine-cosine generator [Mitra,1998]







- Cosθ determines the output frequency
- Example, $\theta = 20^{o} \Rightarrow \cos\theta \cong 0.94$
- Note: peak amplitude of $S_1 > 1$!

Frequency Generator – Analysis

$$f = \frac{clk \cdot \theta}{360}$$

- Small f => small θ => Cos θ => 1

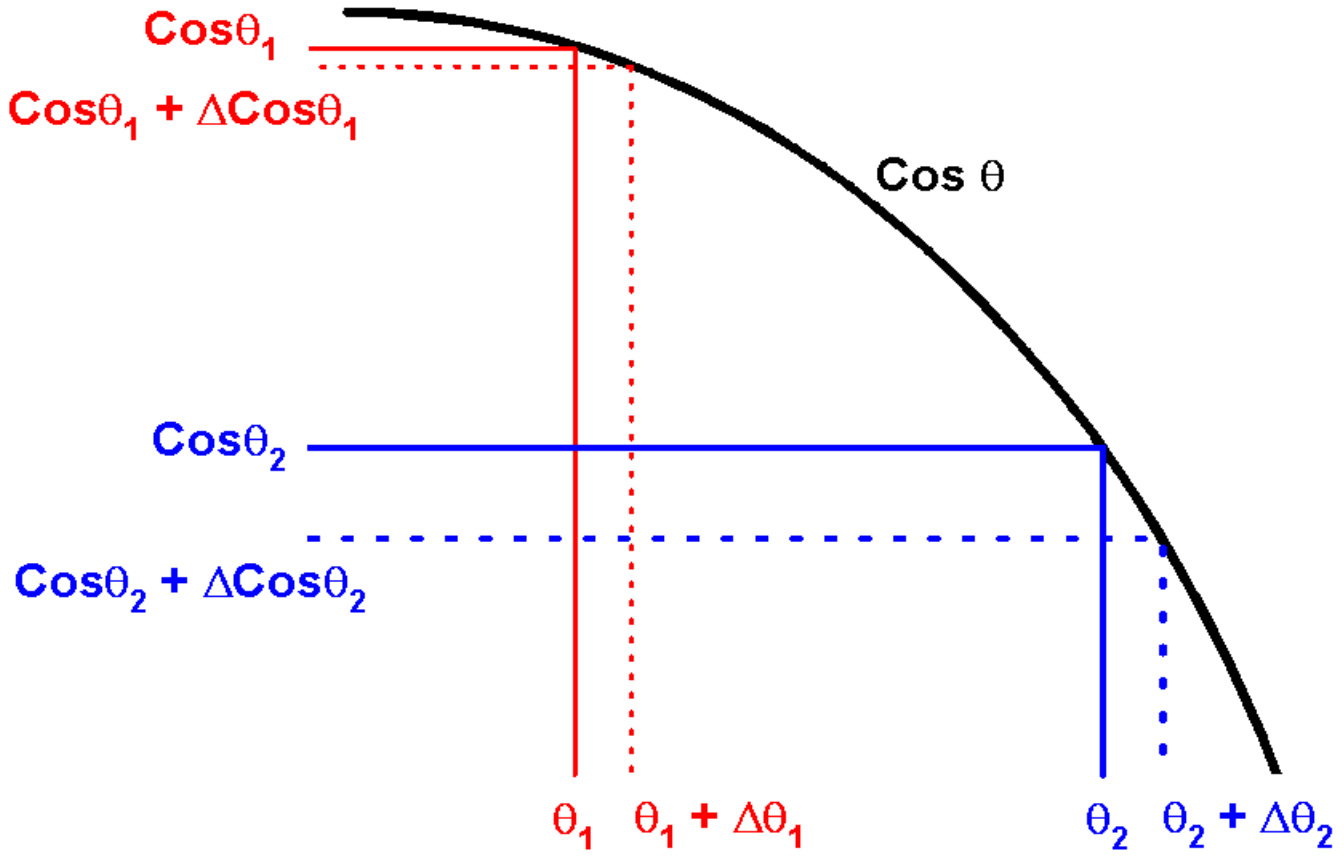
$$Peak\_Amp = \frac{Cos\theta + 1}{Sin\theta}$$

- As Cos θ => 1, Peak_Amp => ∞

  ∴ Peak_Amp set by minimum output frequency (f_{min})

Frequency Generator – Resolution of Cosθ

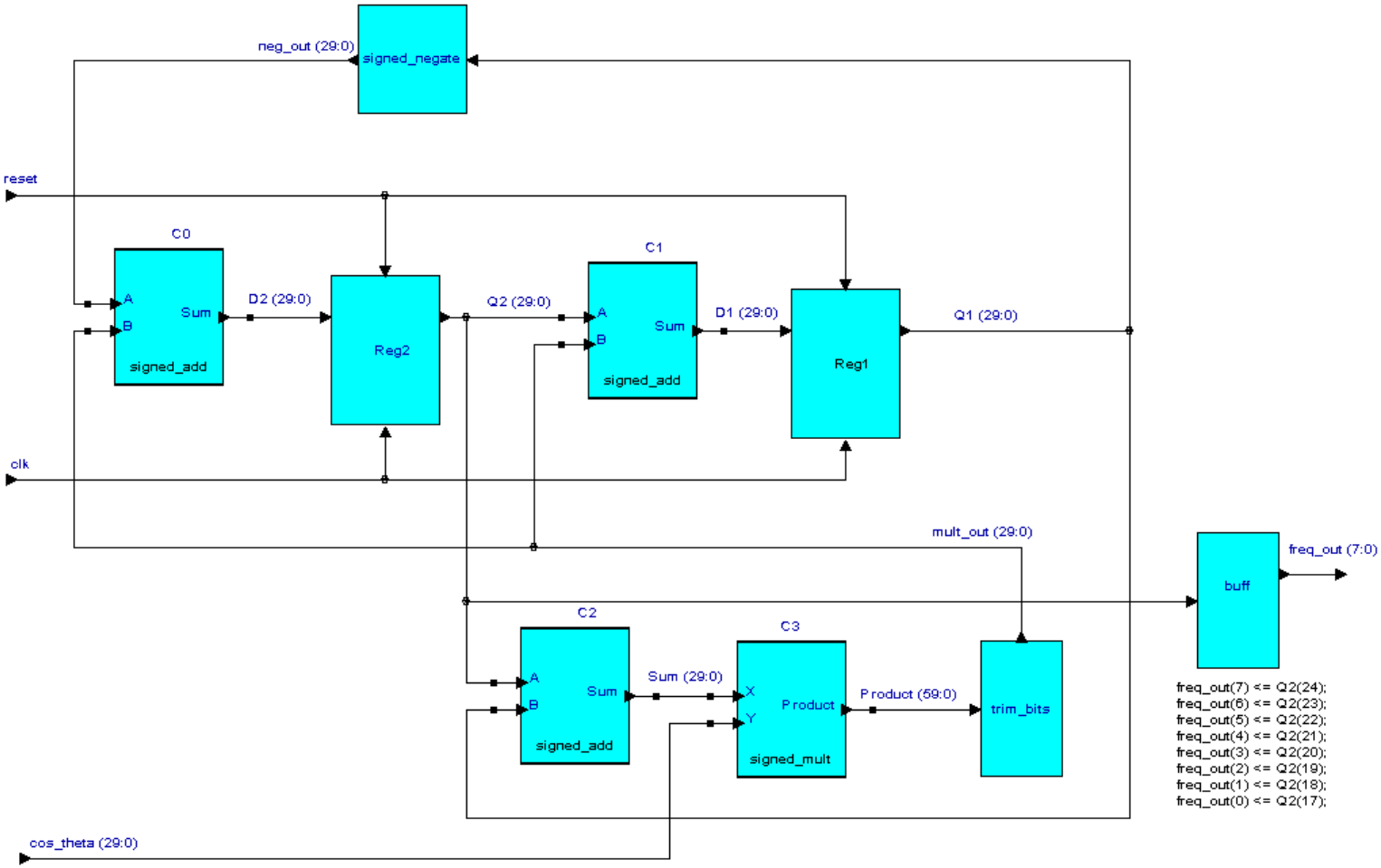
- Set by Δf (=> Δθ) and the minimum output frequency (f_{min})



Frequency Generator – Design Specification

- Frequency range: 200kHz – 500kHz
- Frequency resolution of 2.5Hz
- Sample rate (clk) = 10MHz
- Phase error < 10^{-9} (set by phase error of the sample clk)
- Fixed point arithmetic

Frequency Generator Design



Frequency Generator – Integer Bits Calculation

- Integer bits used to represent Peak_Amp
- Peak_Amp maximum at f_{min}

$$Peak\_Amp = \frac{Cos\left(\frac{360 \cdot f_{min}}{clk}\right)+1}{Sin\left(\frac{360 \cdot f_{min}}{clk}\right)} = \frac{Cos\left(\frac{360 \cdot 0.2}{10}\right)+1}{Sin\left(\frac{360 \cdot 0.2}{10}\right)} = 15.9$$

- => 5 bits

Frequency Generator – Fraction Bits Calculation

- Set by the required resolution of Cos θ

$$Cos\theta\_bits = \log_2\left[\frac{1}{Cos\left(\frac{f_{\min}\cdot 360}{clk}\right) - Cos\left(\frac{\left(f_{\min}+\Delta f\right)\cdot 360}{clk}\right)}\right]$$

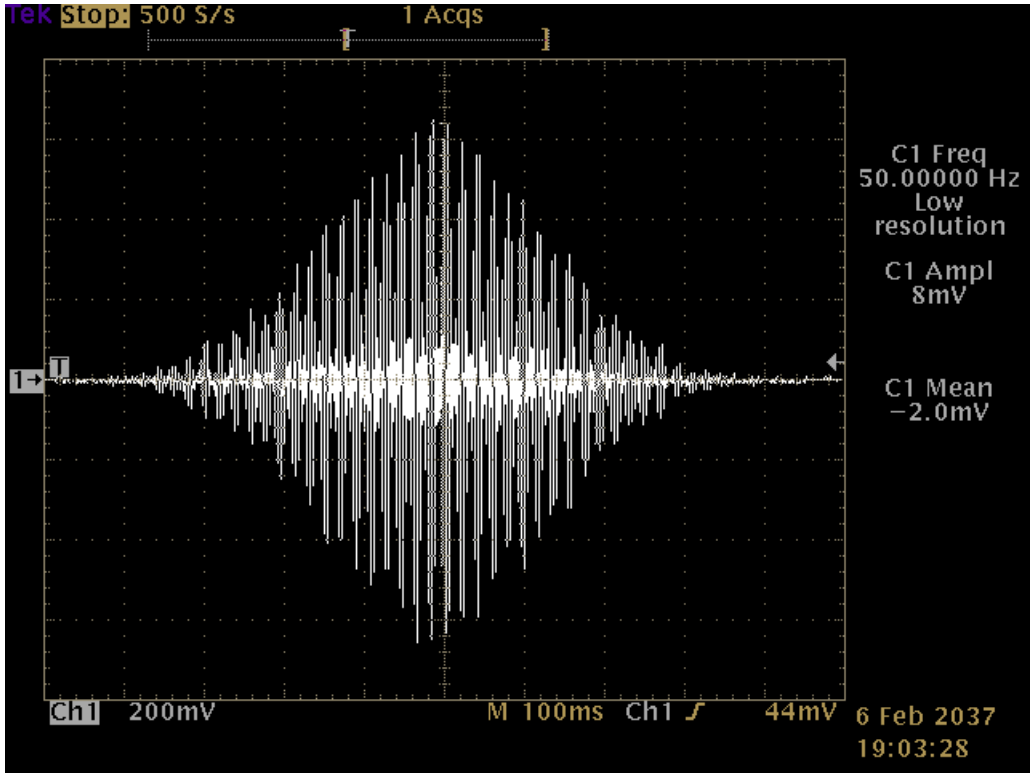
$$Cos\theta\_bits = \log_2\left[\frac{1}{Cos\left(\frac{0.2\cdot 360}{10}\right) - Cos\left(\frac{\left(0.2+0.0000025\right)\cdot 360}{10}\right)}\right] = 23$$

Frequency Generator – Fixed Arithmetic Bits

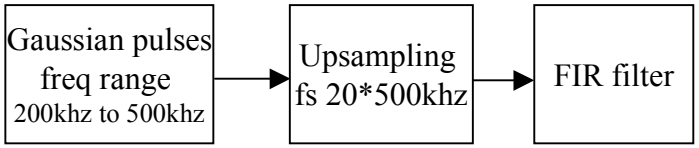
- Integer : 5 bits
- Fraction : 23 bits
- + 2 bits to minimise arithmetic overflow errors
- TOTAL : 30 bits

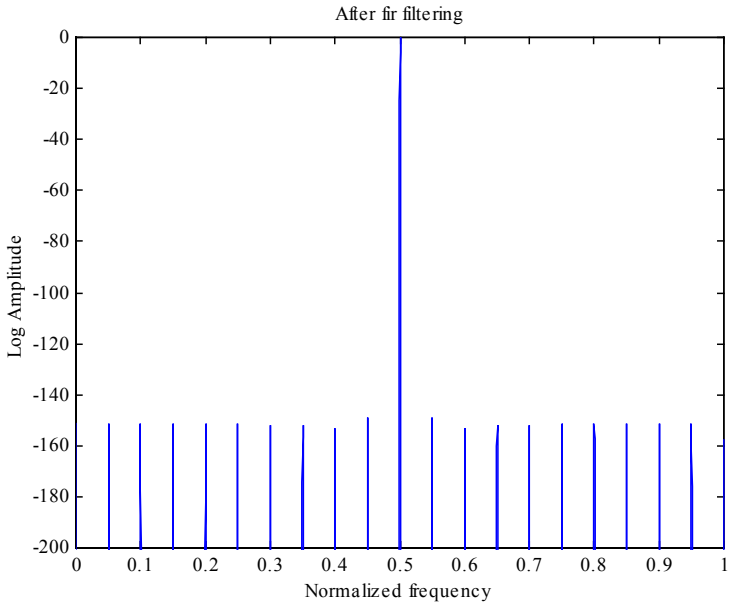
Frequency Generator – Hardware Implementation

- Have currently implemented frequency generator (of lower range), 8bit Gaussian generator, and 8x8bit multiplier in ~15000 gates

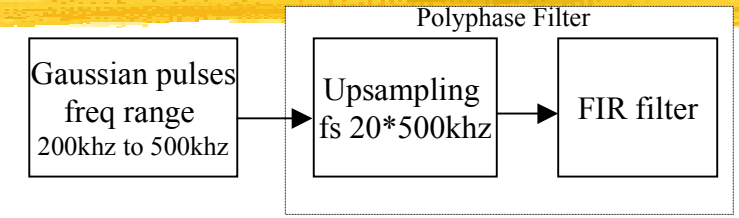


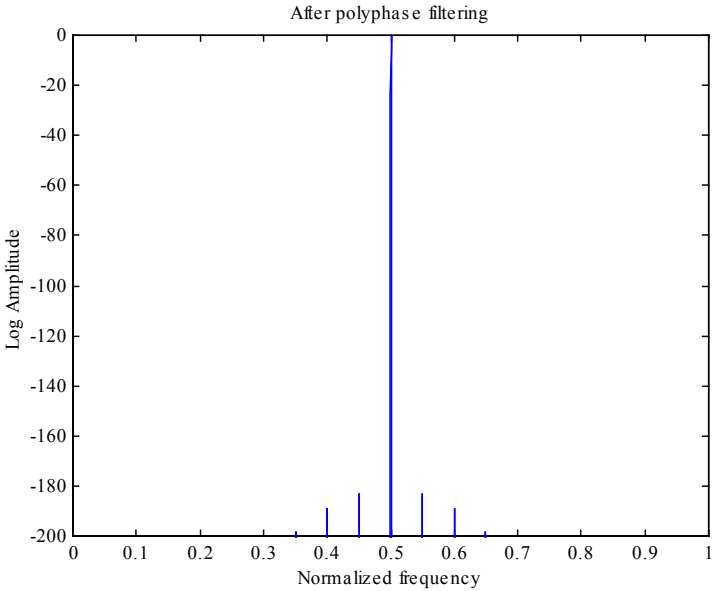
FIR vs Polyphase Filter Implementation





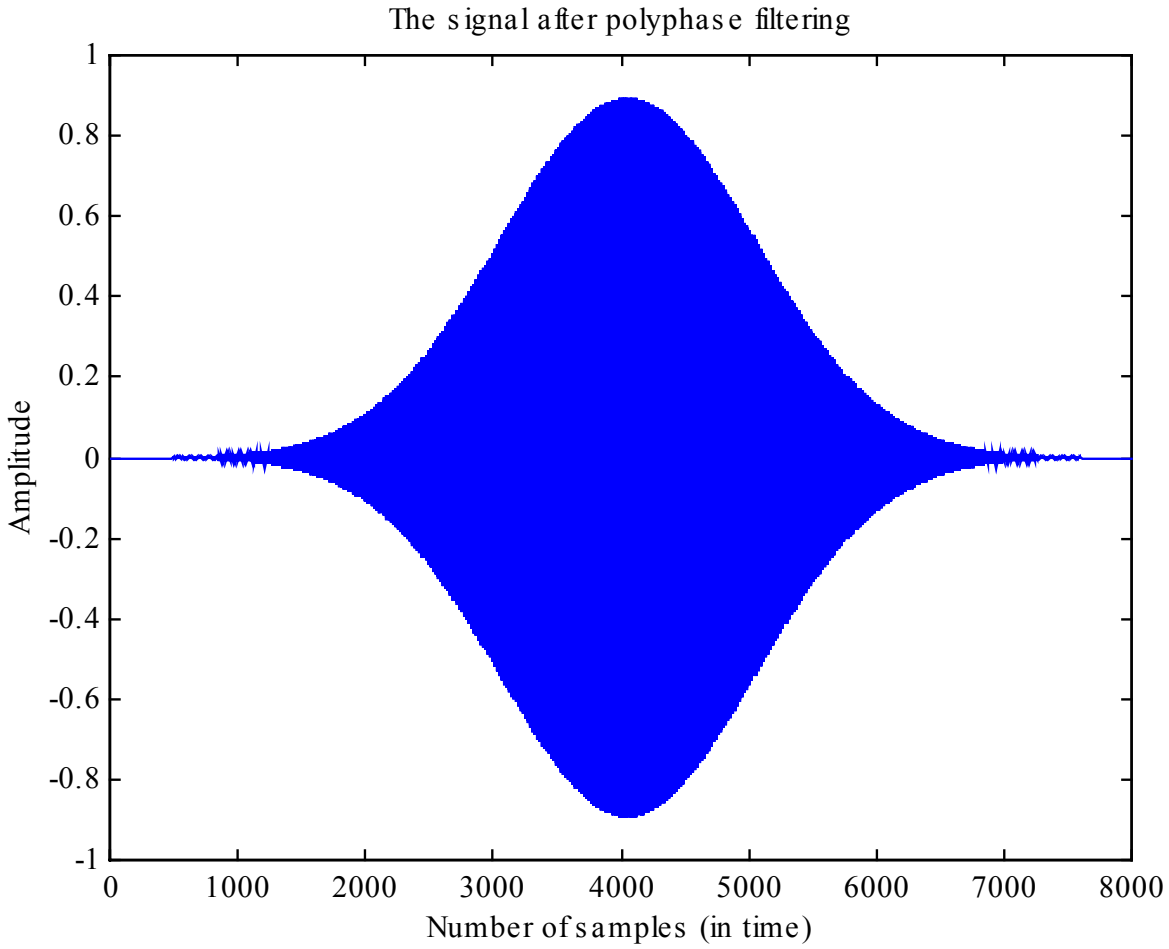
- Spectral copy of upsampled Gaussian pulse after FIR filtering
- Filter length 100 taps
- Filter operates at output sampling rate





- Spectral copy of Gaussian pulse after polyphase filtering
- Filter length 100 taps
- Only the output section of the Polyphase Filter runs at the full sample rate

Poly-phase Filter – Matlab Simulation



Hardware Implementation of Poly-phase Filter

- Have currently implemented an 8bit, 5-tap poly-phase filter in FPGA
- Implementation requires ~2000 gates
- Estimate ~40000 gates for 100 tap filter

Discussion (1)

- Easier to build and maintain - cost
- Less space
- FPGAs are reconfigurable
  - Tailor system configuration to suit any particular mission
    - e.g. infinite control over receiver bandwidth

Discussion (2)

- 16 (or 20) receivers allows each to be recorded and stored independently – 16 (or 20) times data rate and storage
- Allows for beam forming in post processing using polyphase filtering to form the beams and interpolate within the beam
- IFF we illuminate the whole of the 52 degree field of view we have all beams available all the time
- Drawback is we would need 10kW transmitters

Conclusion

- Design of digital implementation of a SuperDARN transmitter stage

  (excluding power amp – of course)
- Shown feasibility of this design in a proof of concept implementation
- Understand the issues involved in the implementing the variable RF synthesiser

Future Work

- Investigate tradeoffs between number of bits in the frequency Synthesiser and a variable clock rate
- Implement 100 tap poly-phase filter in FPGA
- Investigate a Receiver implementation