Designing with DDS

A high speed solution for portable and handheld applications

Jeff Keip - Analog Devices
&
David Askins – Avnet Electronics

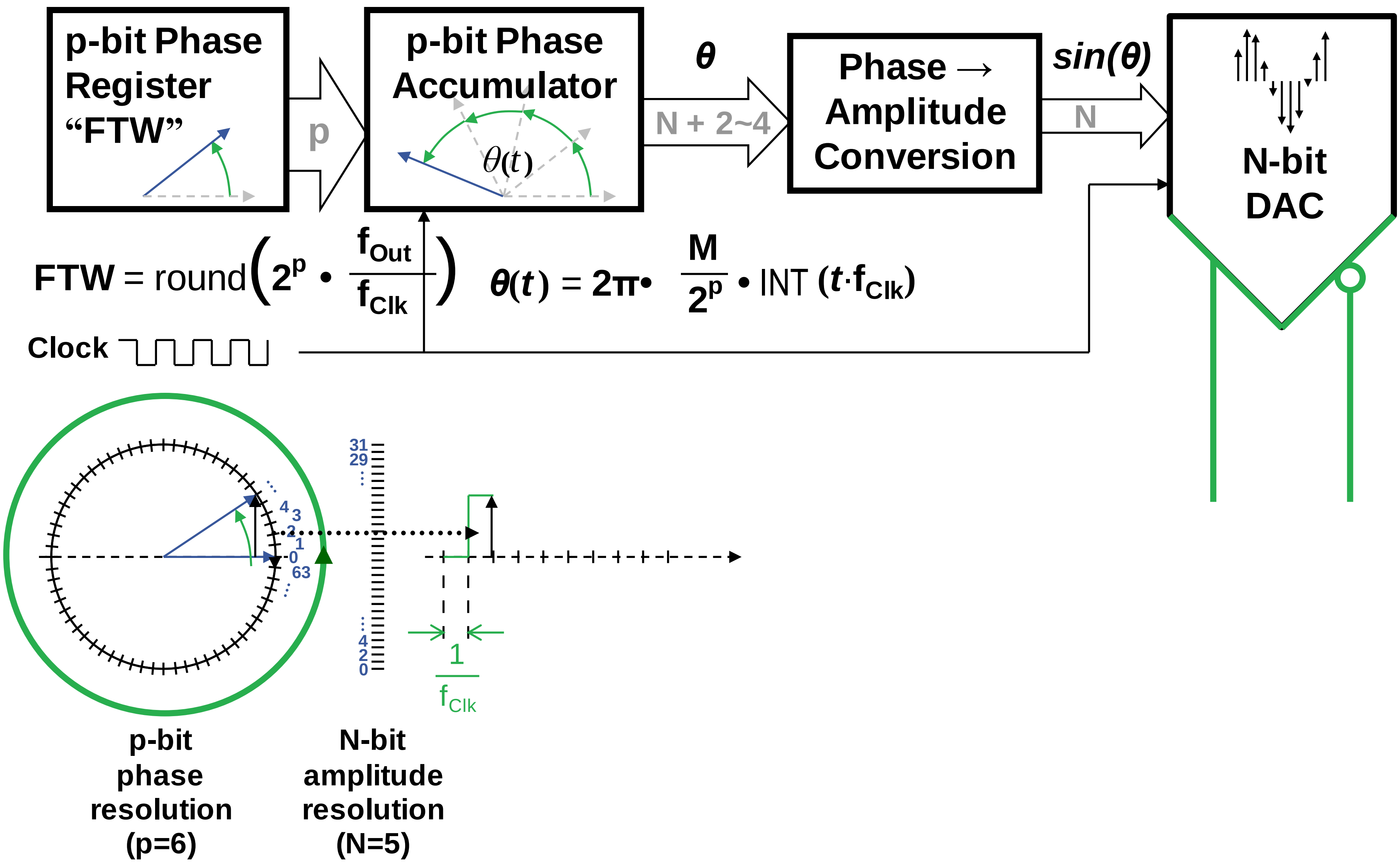
Agenda

- Basic DDS review
  - DDS engine
  - DDS features
- AD9913 – ultra low power DDS
- Comparing stand alone DDS vs. FPGA embedded DDS engine
  - Ease of implementation
  - Performance
  - Cost
- Programmable Modulus explained

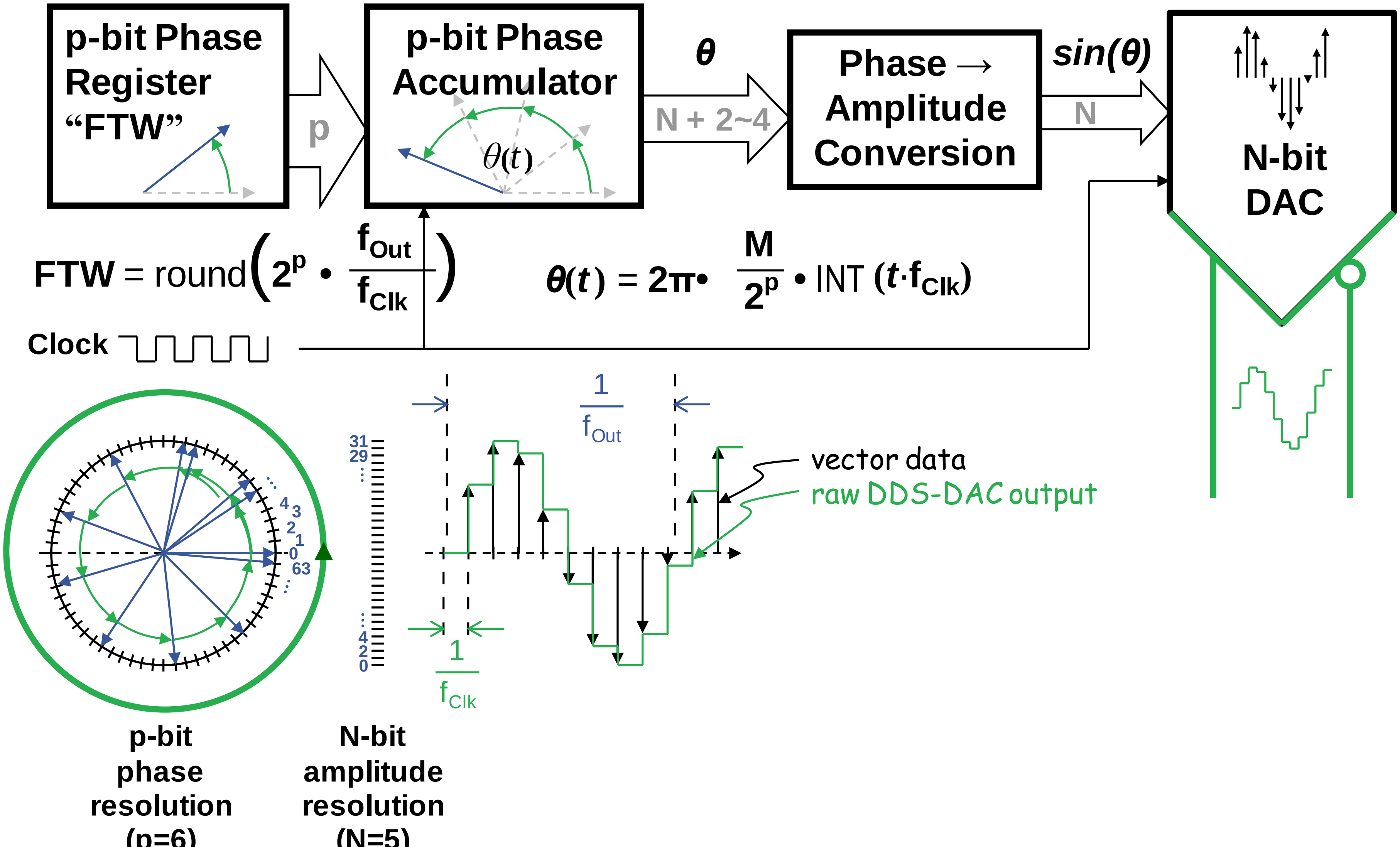
Quick Review of DDS

- **Direct Digital Synthesis is:**
  - A DIGITAL technique for generating sine waves from a fixed-frequency clock source
- **By using DDS:**
  - The sine wave FREQUENCY is digitally tunable to sub-Hertz resolution
  - The sine wave PHASE is digitally adjustable
  - There are NO ERRORS from drift due to *temperature* or *aging* of components
  - The synthesizer is AGILE; frequency and phase changes are made quickly with near-zero settling time
  - Frequency, phase and amplitude ramps are easy to implement

Basic DDS System – p-bit DDS with N-bit DAC



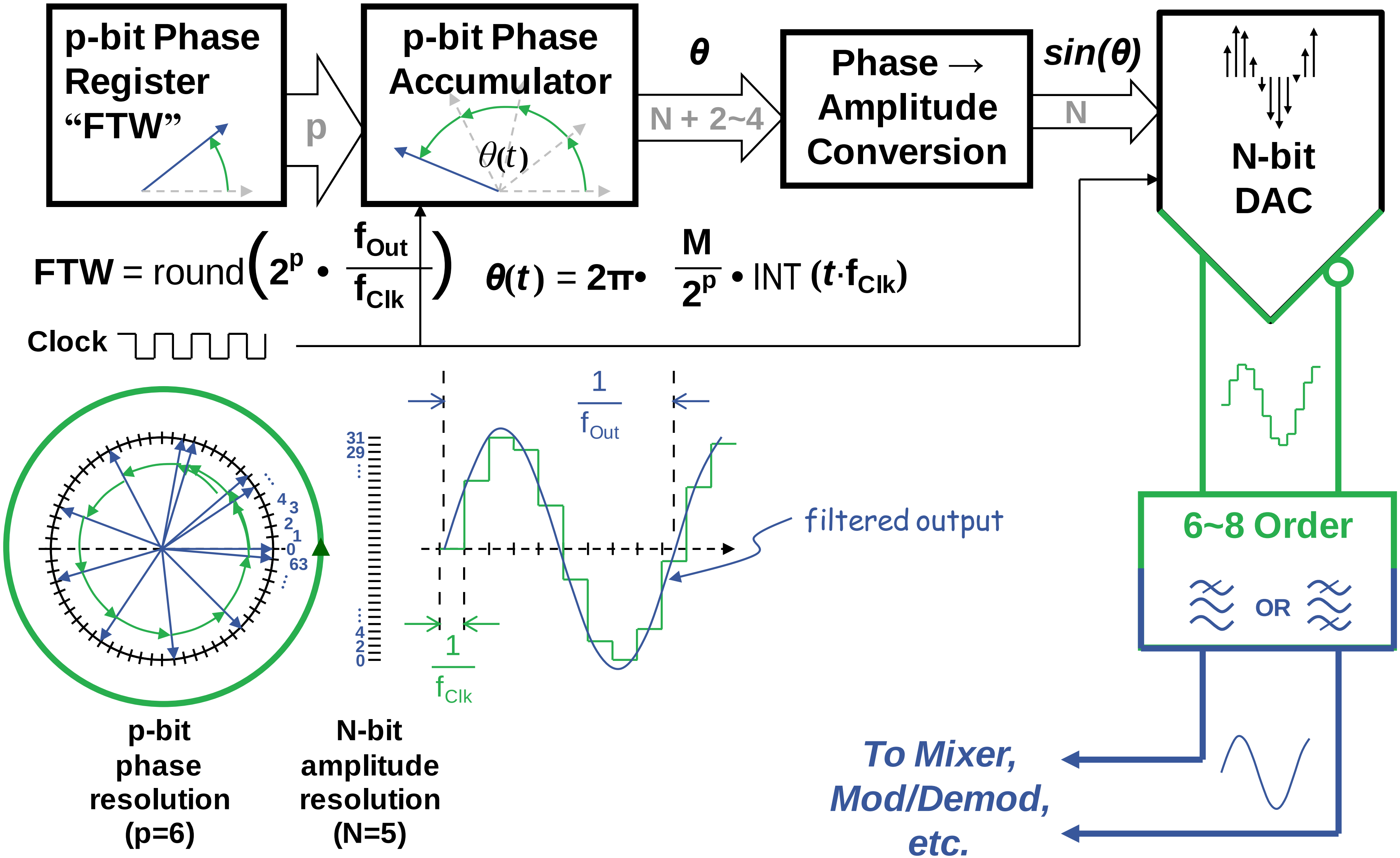
Basic DDS System – p-bit DDS with N-bit DAC



$$FTW = \text{round}\left(2^p \cdot \frac{f_{Out}}{f_{Clk}}\right)$$

$$\theta(t) = 2\pi \cdot \frac{M}{2^p} \cdot \text{INT}(t \cdot f_{Clk})$$

Basic DDS System – p-bit DDS with N-bit DAC



$$FTW = \text{round}\left(2^p \cdot \frac{f_{Out}}{f_{Clk}}\right) \qquad \theta(t) = 2\pi \cdot \frac{M}{2^p} \cdot \text{INT}\,(t \cdot f_{Clk})$$

Useful DDS Features

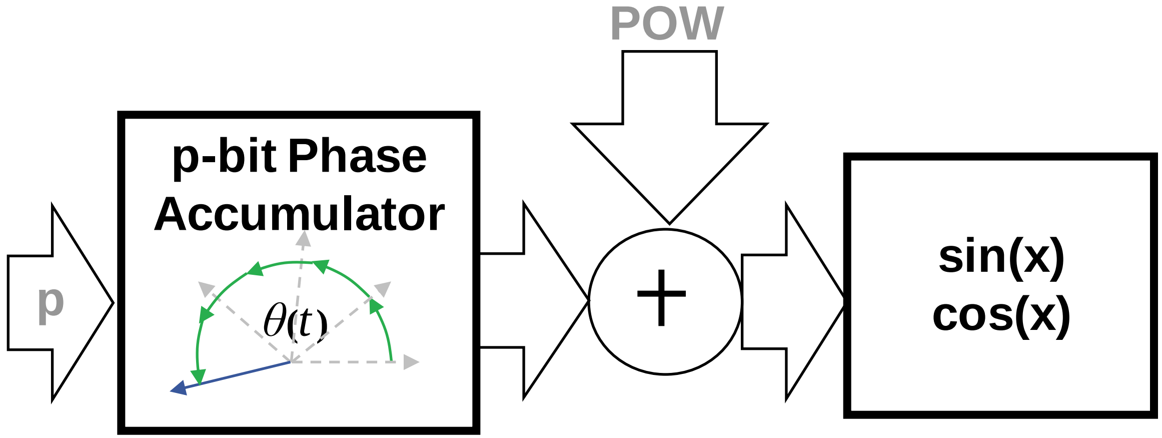
- **Frequency Tuning Resolution**
  - Finer tuning capability than even fractional N PLLs
- **Phase Offset Word**
  - Phase tuning implemented digitally to .022° resolution
- **Auxiliary (Sweep) accumulator**
  - Enables Frequency/phase/amplitude ramping and Programmable Modulus
- **Amplitude Scaling Factor**
  - Digital magnitude control on output
- **Profiles/Shift Keying**
  - Extremely low settling times with no ringing
- **RAM**
  - One way of implementing non-linear sweeps and shift keying
- **Matched Latency**
  - Frequency, phase, and amplitude changes all affect the output simultaneously
- **Clear Phase Accumulator**
  - Can establish known phase on F/P/A changes

Phase Tuning Resolution

- **POW (Phase Word -- K bits)**
  - **Register value sets resolution**

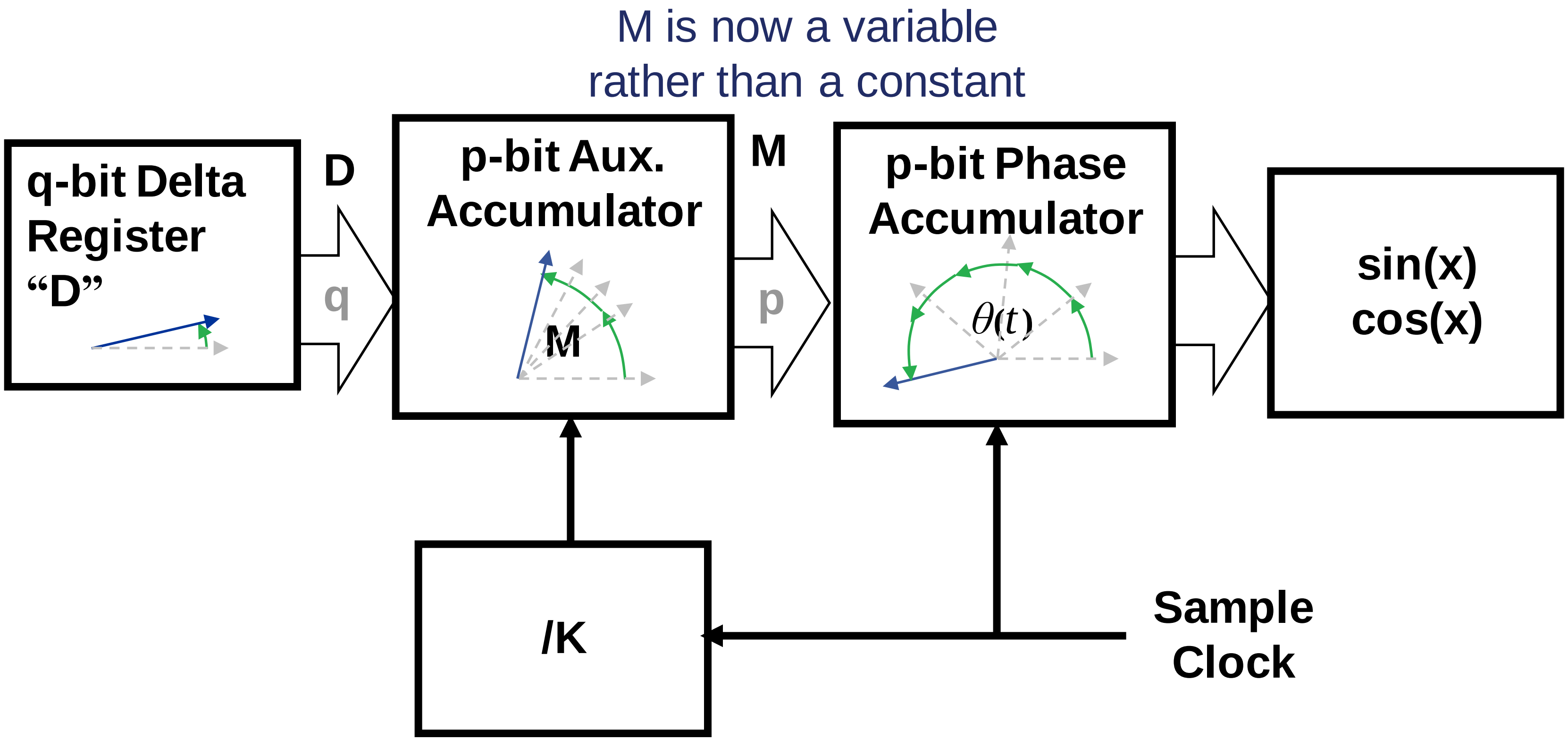
phase equation: $\Phi_{out} = \dfrac{POW \times 360°}{2^K}$ degrees

$= \dfrac{POW \times 2\pi}{2^K}$ radians



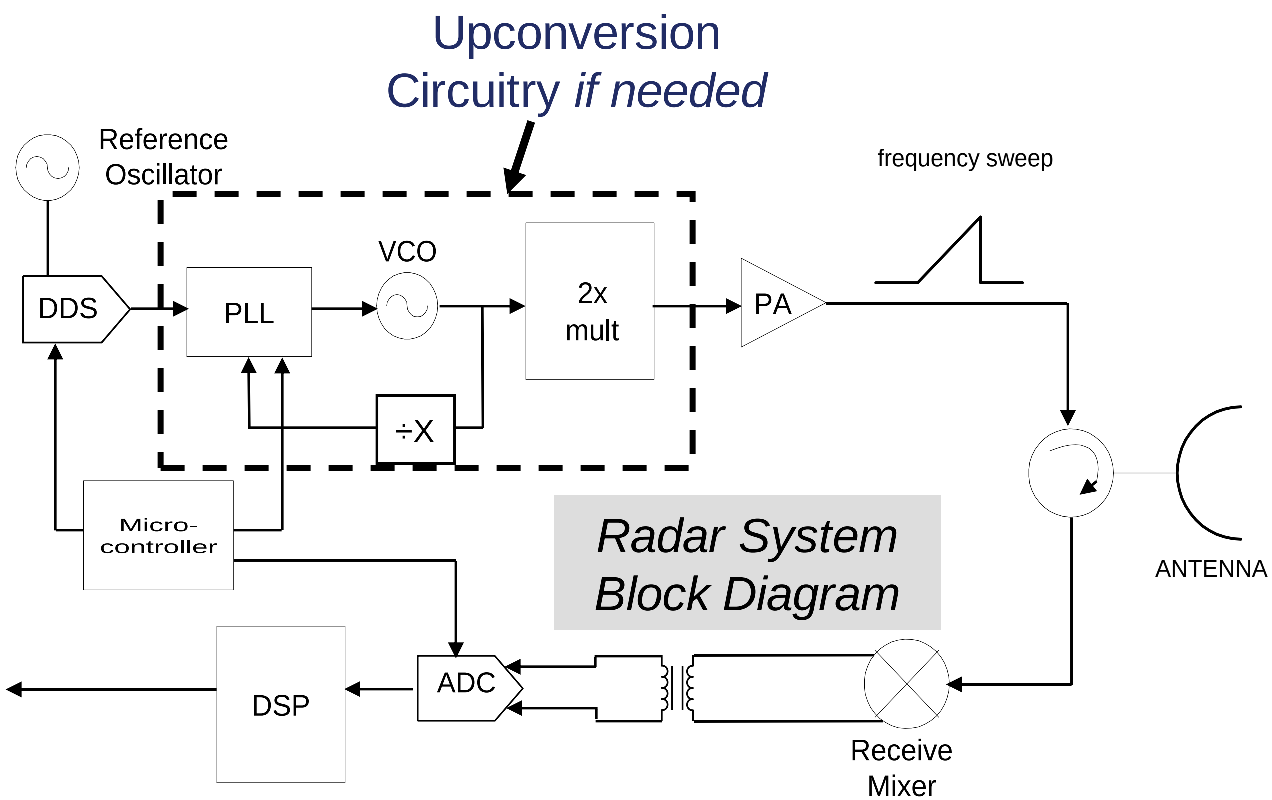
Auxiliary Accumulator

- Ease of implementation for frequency sweeping makes DDS good option in radar system



Auxiliary Accumulator: Frequency Sweep

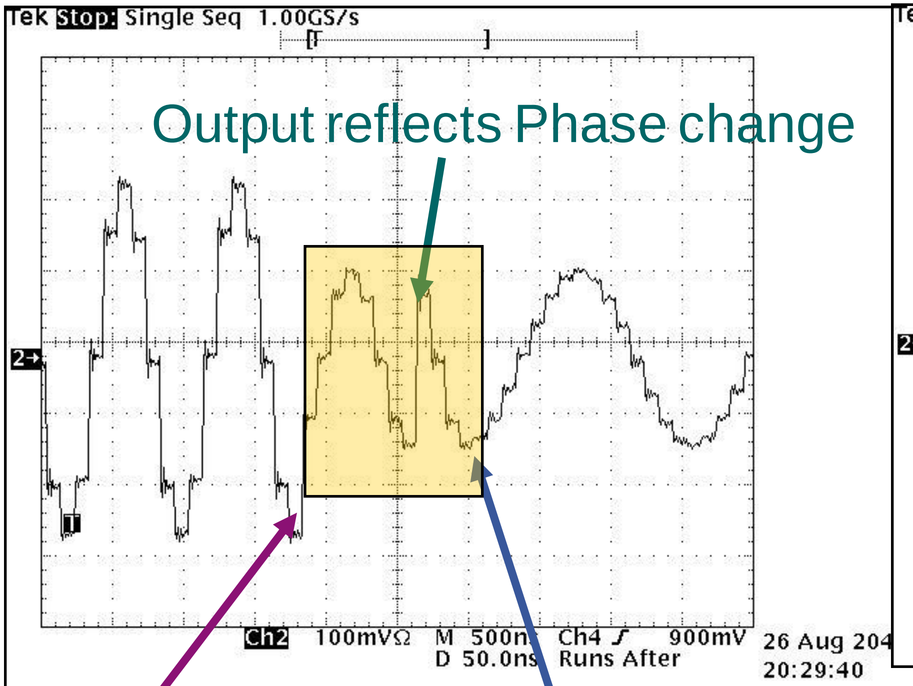
- Ease of implementation for frequency sweeping makes DDS good option in radar system

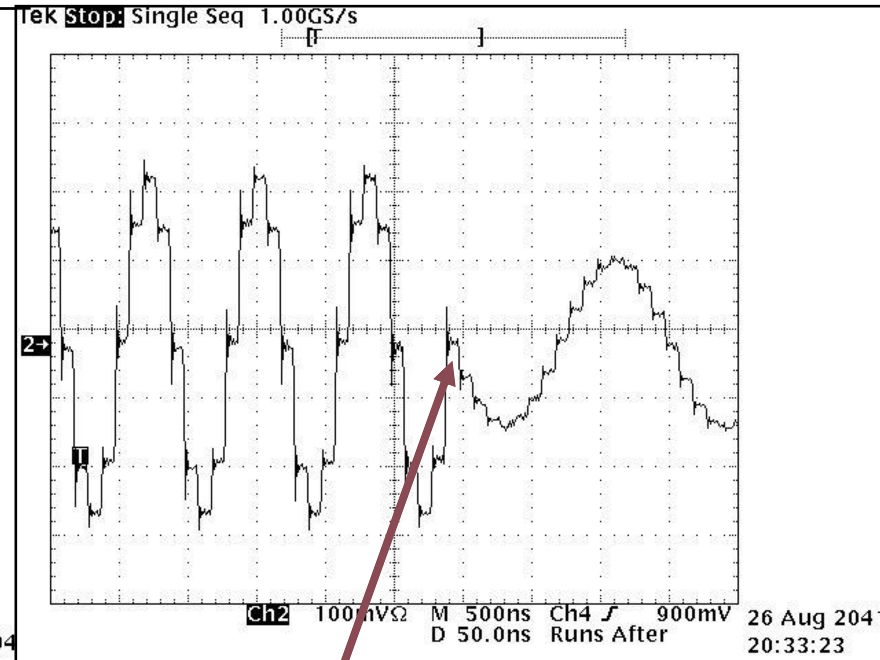


Matched Latency

Unmatched

Matched



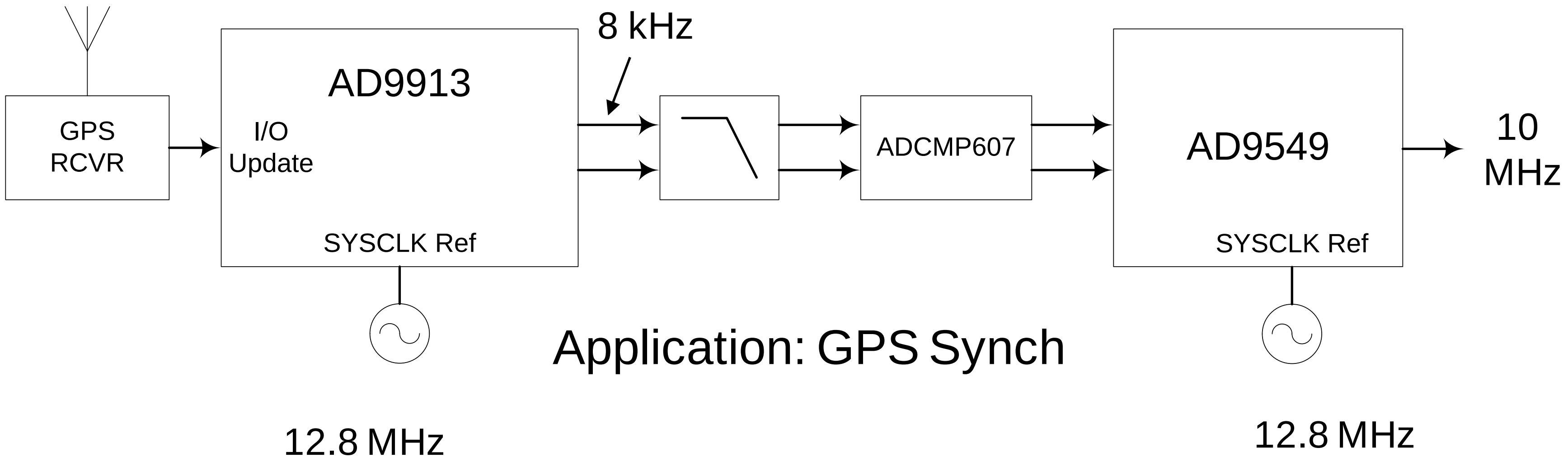


Output reflects Amplitude change

Output reflects Frequency change

Output reflects A, Φ, and F changes simultaneously

Application Example – Clear phase accumulator



- Clear Phase accumulator function enables system to reset the phase of the DDS to 0° easily
- May be made automatic when changing profile

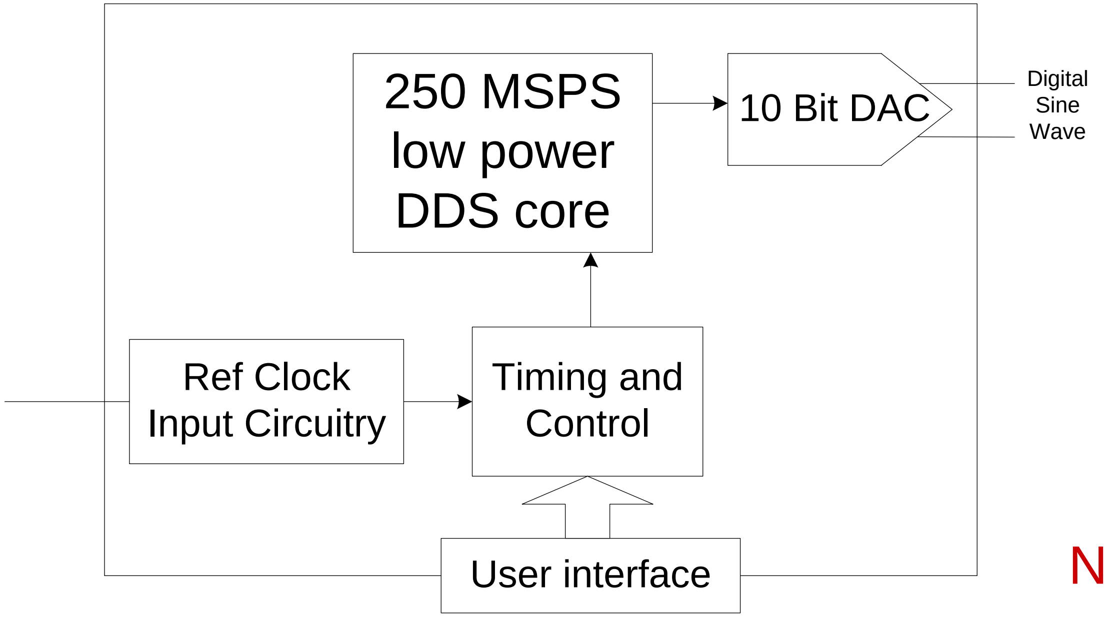
Useful DDS Features

- **REFCLK multiplier & PLL (Agile*RF*)**
  - Allows for output frequencies higher than the reference
- **Multi-chip Synchronization**
  - Digital and precise
- **Zero Crossing Bit**
  - Eases system synchronization efforts
- **Comparator**
  - Allows for square wave output
- **Angle to Amplitude Bypass**
  - Digital ramping output instead of sinusoid

AD9913

Primary Design considerations

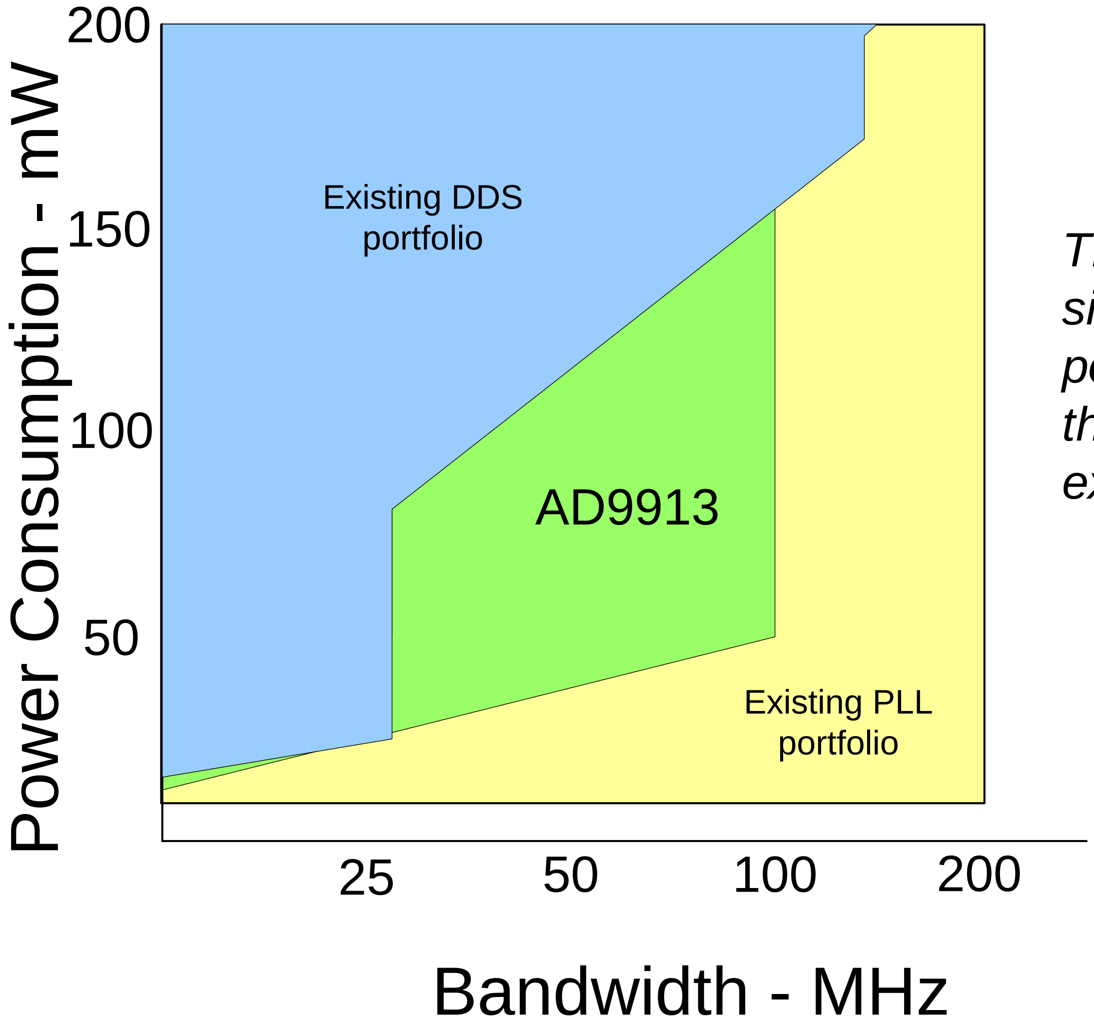
- **Low Power** <50 mW at full speed
- **Low Price** Price-equivalent to comparable DAC
- **Very small** 5x5 32 pin LFCSP



Compare to FPGA embedded DDS approach

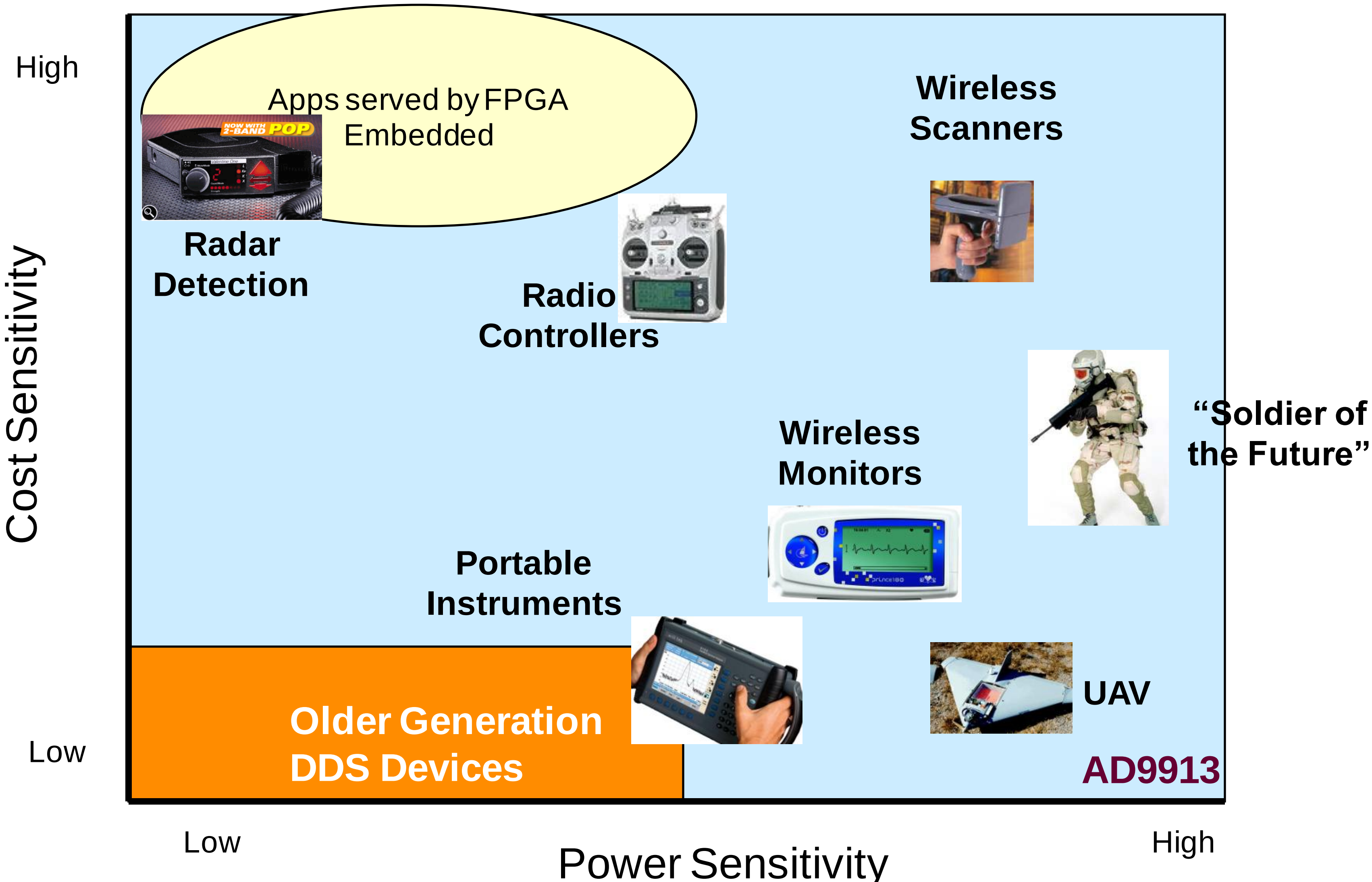
Lower power
Easier to design
Cost effective
No flexibility in DAC selection
Not quite as fast

The Greening of DDS for IF generation

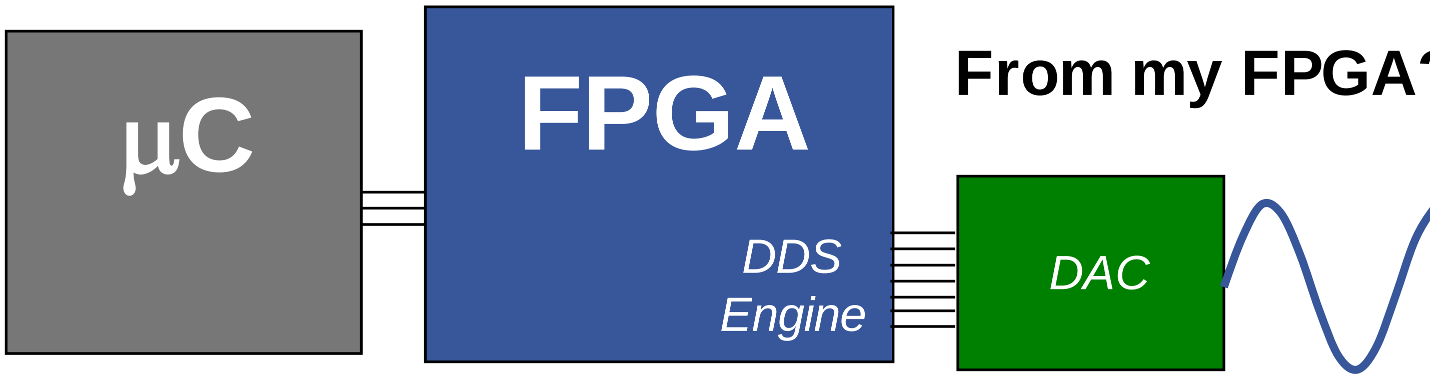


The AD9913 cuts a significant swath in the power/frequency gap that exists between existing DDS and PLLs

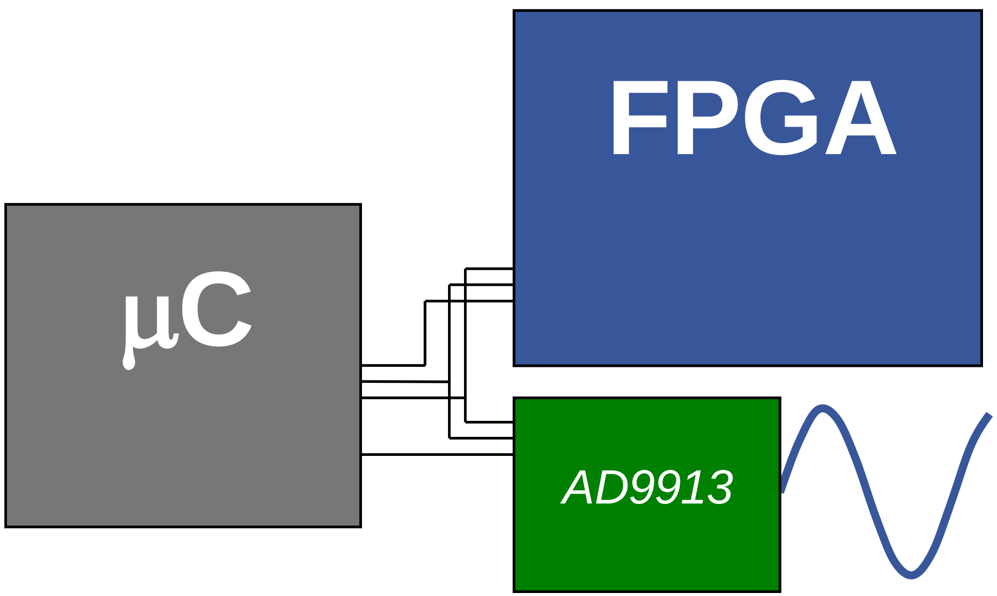
AD9913 application examples



“Where should I get my DDS function?”



From my FPGA?



Or on my DAC?

Key factors when implementing a DDS in a FPGA

- Ease of implementation
- Amount of resources used, i.e. memory cells or logic blocks used.
- Performance
- Cost

Ease of Implementation

- Generating a DDS engine in an FPGA is straight forward and simple
- Utilize Core Gen tool to develop the engine
  - GUI interface makes it easy
  - Input the desired DDS requirements and parameters
  - F_{OUT}, modulation scheme FSK, ASK…, SFDR level, etc
  - Build or compile the core and load it into the FPGA
- The Core Gen tool will build the DDS engine accordingly

Amount of resources used

- Resources in an FPGA are memory cells
- FPGA logic implementation typically requires 3x to 4x the silicon content for the same IC implementation
- Phase to amplitude conversion in a FPGA is done with a lookup table
  - The table is sized according to desired SFDR level
  - This can make the table very deep requiring a very large amount of resources
  - The AD9913 utilizes a real time algorithm for the phase to amplitude conversion

Performance of the FPGA DDS

- The DDS engine in a FPGA can be very similar to that of a stand alone DDS such as the AD9913
  - Spurs can be set at a desired level
  - Noise floor can be set to a desired level
  - Similar modulation schemes can be chosen etc.

- All this comes at the price of increased resources or memory cells and increased cost.

Performance of the FPGA DDS continued

- Now let’s look at some major design considerations
- The FPGA DDS is the digital DDS engine only, an external DAC is required in order to generate the desired analog output
- This is critical when implementing a DDS design for a couple of reason
  - The interface between the DDS engine and the DAC
    - The interface can be a source of noise generation and reduced performance if not done properly
  - The choice of DAC
    - Choice of the DAC can greatly affect the overall system performance
    - SFDR and noise floor can degrade due to a low performance DAC

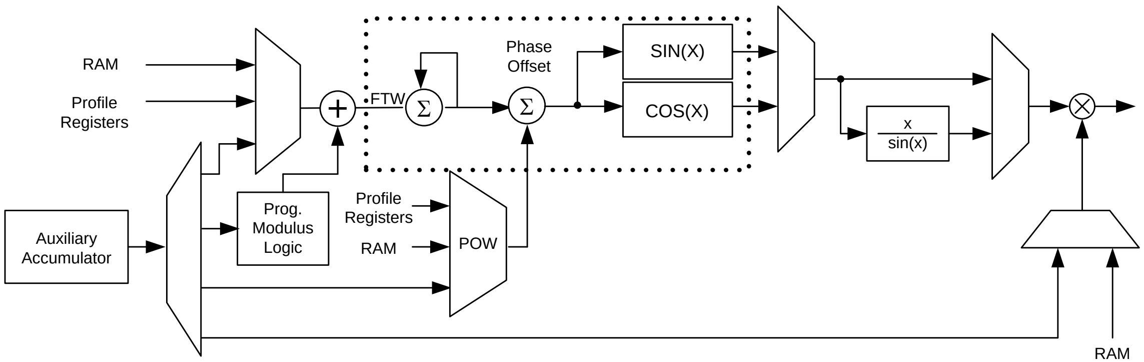
Performance of the FPGA DDS continued

- With the AD9913 the DAC function is integrated onboard the device
  - The DAC is matched to the DDS core to maximize system performance
- Design requirements are simplified for the engineer

Cost of Implementation

- The cost if implementing a FPGA DDS is higher due to
  - The increased amount of resources required to implement the same IC function, i.e. 3x to 4x silicon requirement
  - Lookup table depth
  - Requires an external DAC

Auxiliary Accumulator: Programmable Modulus



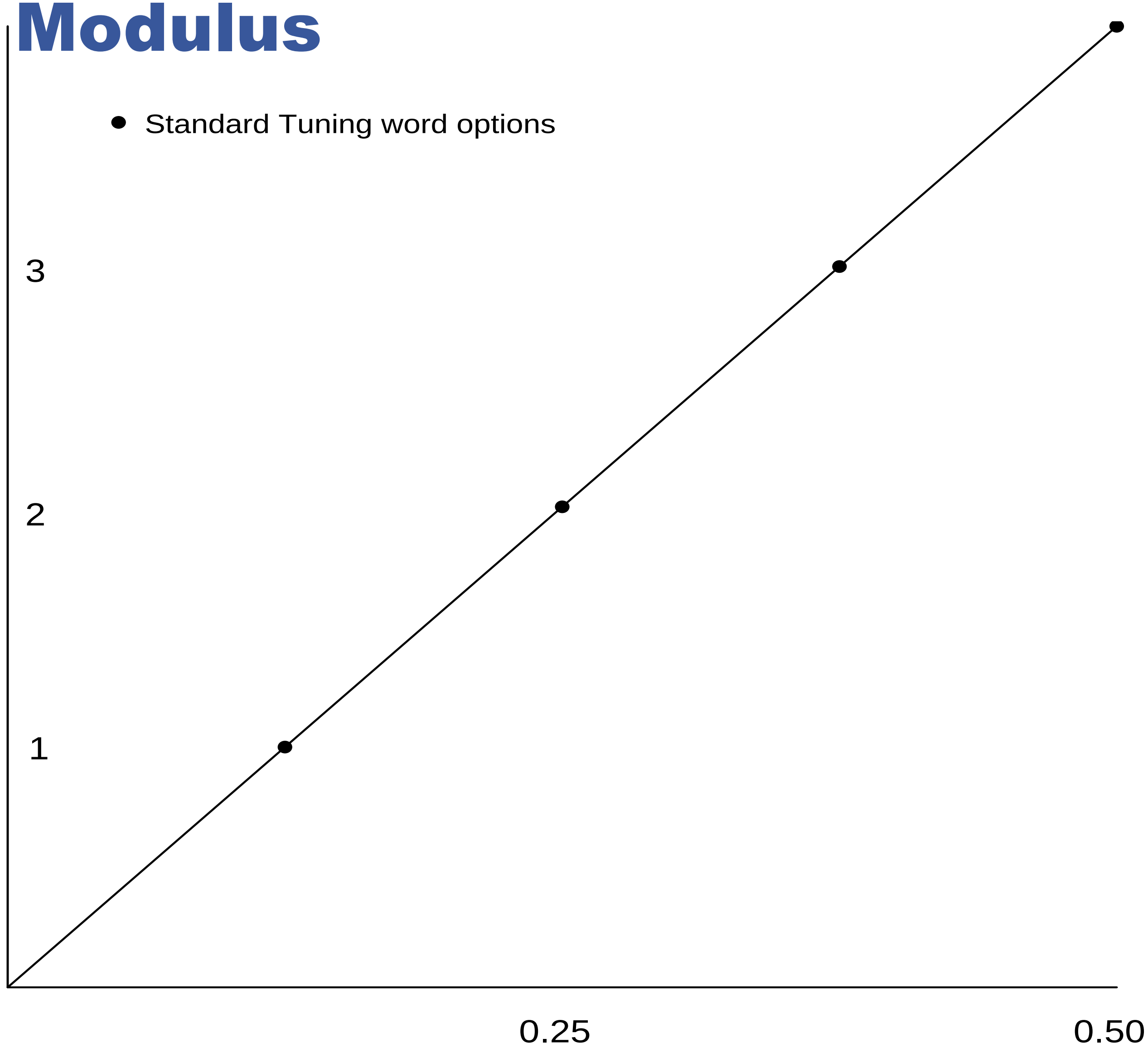
State of the art DDS Core block diagram

- **When Programmable modulus mode is active, The divisor in the DDS frequency equation becomes a user-defined variable**

$$\textit{New DDS frequency equation:} \quad F_{out} = \frac{FTW \times F_{SAM}}{Y}$$

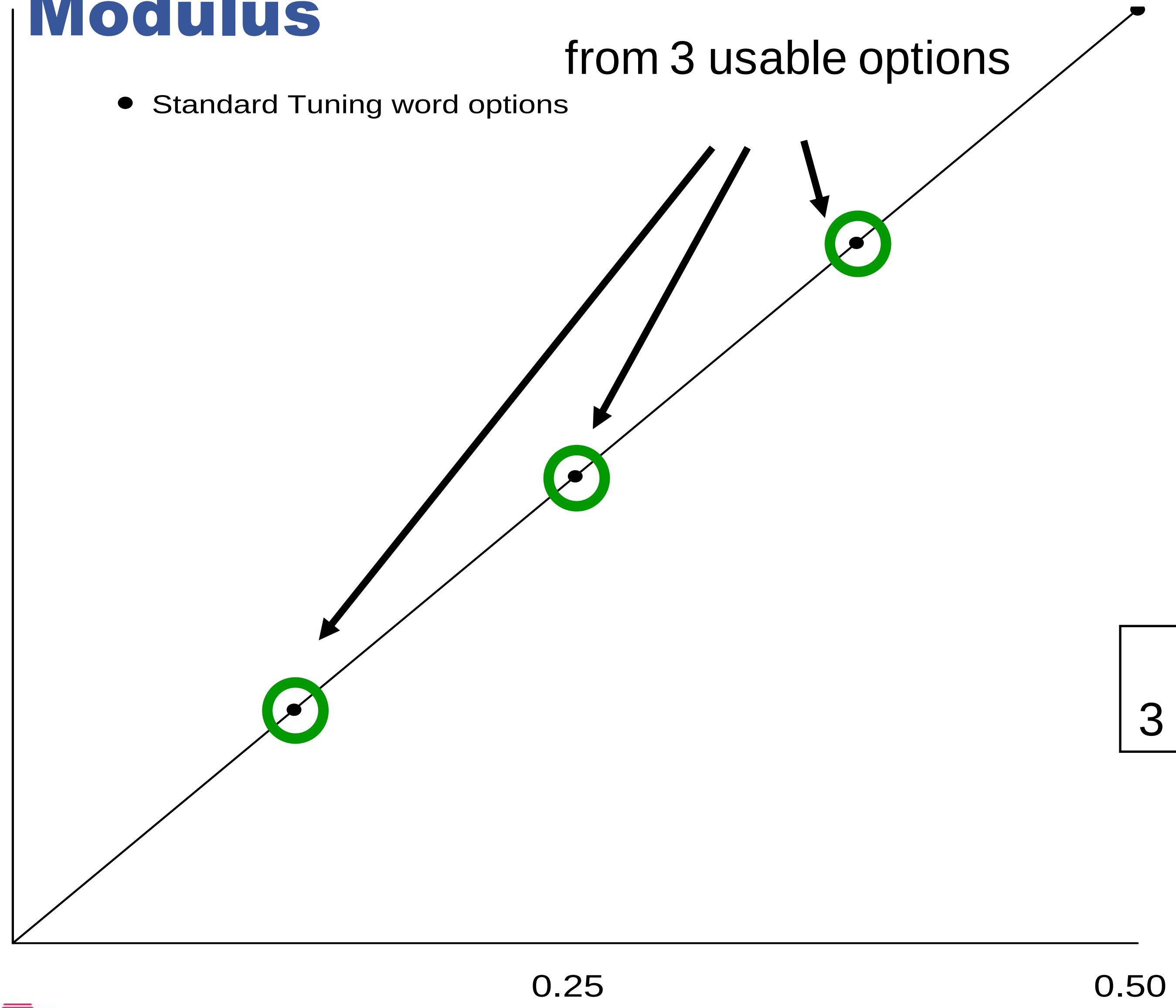
Y is any value between 2·(FTW) and 2^Z, where Z is the # of bits in the linear sweep accumulator, or select values between 2^Z and 2^{X+Z} where X is the # of bits in the phase accumulator

Tuning Resolution effect of Programmable Modulus

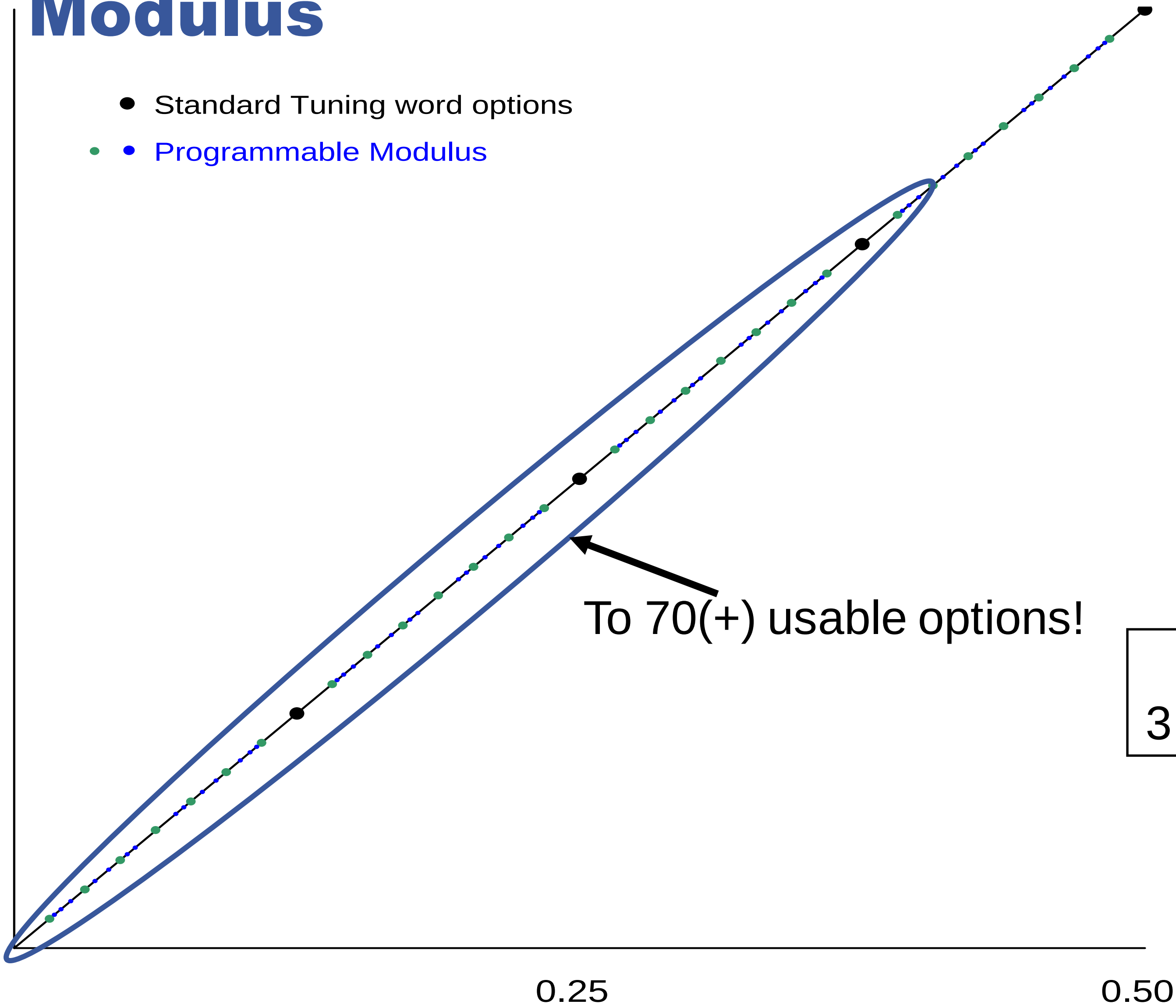


Example using
3 bit accumulators

Tuning Resolution effect of Programmable Modulus



Tuning Resolution effect of Programmable Modulus



Example using
3 bit accumulators

Conclusion

- The introduction of the AD9913 means
  - DDS is now significantly more usable in portable/handheld applications
  - Designers relying on FPGA based DDS engines have a better alternative
  - The frequency tuning benefit of DDS has taken a quantum leap forward thanks to Programmable Modulus