Learning goals

- AnadigmApex FPAA switched capacitor technology
- AnadigmApex silicon components and architecture
- AnadigmApex EDA tools
  - AnadigmDesigner®2
  - AnadigmFilter®
- Generating configuration records
Anadigm® FPAA Solutions - Agenda

- FPAA Technology basics
- FPAA Silicon overview
- Design software overview
- Exercises
- Next Steps – How you can start working with Anadigm programmable Analog
FPAA Switched Capacitor Technology
FPAA ‘Switched Capacitor’ Technology

- The Anadigm FPAA devices are based on switched capacitor technology. There are no resistors or inductors in SC systems as these are replaced by ratios of capacitors.
- SC technology is a sampled analog system and requires that the engineer choose sample clocks for the CAMs. Normally, the sample clock will be 10x or greater than the highest frequency you will be processing.
- It is important to note that sampled analog is not digital!
  - Sampled Analog system quantize only in time while
  - Digital systems quantize in both in time and amplitude
Sampled Analog is not digital!
FPAA Switch Capacitor - Why Switched Capacitor?

**High accuracy: Circuit accuracy typically ± 1%**

- CMOS capacitor tolerance typically ± 1.0%
- Capacitor ratio ± 0.1%

- No drift due to temperature and aging.
- Better Linearity than semiconductor resistors

\[
\frac{V_{out}}{V_{in}} = \frac{-R_2}{R_1} = \frac{-1/f_c C_2}{1/f_c C_1} = \frac{-C_1}{C_2}
\]
AnadigmApex Silicon Overview
Part No. AN231E04-E3-QFNxx
AnadigmApex – Silicon Components

- Fully differential architecture
- Same basic building blocks in all Vortex devices
  - Configurable Analog Blocks (CABs)
  - Input cells and IO cells
  - Output Cells
  - Look up table
  - Successive approximation registers
- Several different flavors that vary in component count and IO capability while all physical Packages are the same size
- Configured via SPI interface either from a Processor or an EPROM
AnadigmApex – Silicon Components
The CAB contains components that are used to build circuits in the AnadigmApex FPAA. These are:

- Eight - 8 bit capacitors
- Two op amps
- One Successive Approximation Register (SAR)
- Interconnection logic switches, and wires
The AnadigmApex – CAB: The role it plays

- The CAB is the most important building block in the AnadigmApex FPAA circuit!
- The CAB components are interconnected with each other and with components in other CABs to build circuit functions. A few examples:
  - Gain Stages
  - Filters
  - Rectifiers
The AnadigmApex – CAB: Number in each device

- The CABs are identical in all Apex devices.
- All Apex devices have 4 CABs
AnadigmApex – I/O Cells
AnadigmApex – Type 1 I/O Cells

- There a total of 4 type 1 I/O pins
- Each cell has 4 pins – a differential pair of inputs and a differential pair of outputs
AnadigmApex – Type 1 I/O Cell Input structure

- Each type 1 I/O cell can be configured a number of ways when in Input mode
  - In bypass mode
  - In Sample & Hold mode
  - With a continuous time low pass filter
  - 2 of the 4 I/O cells have a low offset chopper amplifier option
Each type 1 I/O cell can be configured a number of ways when in Output mode:
- In bypass mode
- In sample & hold mode
- In digital output mode: outputs differential digital signal
- In VMR mode: sets the output to Voltage Mid Rail (+1.5V differential)
AnadigmApex – Type 2 I/O Cells

- There are a total of 3 type 2 I/O pins
- Each cell has 2 pins – a differential pair that can be configured as inputs or outputs
AnadigmApex – Type 2 I/O Cells

- Each type 2 I/O cell can be configured in a number of ways:
  - Input bypass mode
  - Output bypass mode
  - Input control signal mode: 2 independent s/e control signals
  - Output control signal mode: 2 independent s/e control signals
  - VMR mode: output Voltage Mid Rail (+1.5V)
Anadigm Apex – Look Up Table
AnadigmApex – Look Up table

- Can be used to
  - Generate Arbitrary Periodic waves
  - Gain profile for compressors and Voltage controlled gain stages
  - Arbitrary transfer function (e.g. sqrt x, 1/x)

- Size 256 bytes
- Can be dynamically reconfigured
- All devices have one look up table
AnadigmApex – SARs
AnadigmApex – SARs (Successive Approximation Registers)

- Used for transfer
  - Arbitrary transfer function
  - Multiplication and division
- One SAR in each CAB
Flavors of Anadigm Devices

- There are 2 different AnadigmApex devices that vary based on
  - Dynamic or Static Configurability
Anadigm Part numbers

**ANx3xExx**

- **Anadigm Header**
- **E, A letter meaning FPAA devices**
- **Number of CAB’s**
- **Third generation FPAA**
- **0** Fixed input and outputs, ADC internal only
- **1** Flexible I/O + ADC CAM
- **1 Statically reconfigurable**
- **2 Dynamically reconfigurable**
AnadigmApex (3.3volt) Architecture

- OpAmps contain an Input offset voltage “auto-nulling” feature. (I/O and core OpAmps)
- SPI configuration interface enables software control
- dualSRAM based configuration for real time state changes and seamless control over analog parameters
- Four type1 “featured” I/O cells, each can be independently powered down or configured as
  - single-ended or differential
  - an independent differential gain stage
  - differential input filter
  - input or output sample and hold
  - a bypass wire or digital output
- Three (type2) simple differential I/O cells.
- One chopper stabilized gain stage (G <= 60dB), available to use with Type1 or type2 I/O cells
- Two logic/control signal outputs
- Clock management providing 6 non-overlapping internal clocks, two with variable phase delay
- Look Up Table for arbitrary waveform generation
- Rich pre-built (CAM) library

---

Four Configurable Analog Blocks (CABs) controlled by a switch capacitor architecture each containing:
- 2 differential 50MHz op-amps
- 1 differential comparator
- 1 SAR based ADC
- 8 programmable capacitors
Configuring the Apex devices

- Configuration data is transferred via standard SPI interface
- The Apex device can self load as an SPI master from an SPI EPROM
- Or it can be loaded as an SPI slave from a processor
- Multiple Anadigm Apex devices can be loaded from the same EPROM or Processor
Configuring the Apex devices

Loading from a SPI PROM

This example assumes internal pull-ups are enabled for ACTIVATE & CFGFLG.
AnadigmApex EDA tools
- **AnadigmDesigner®2**
  - Easy-to-Use
  - Intuitive “drag-and-drop” user interface
  - Standard Windows interface with tool bar and menu selection items
Configurable Analog Modules (CAMs)

- The fundamental software component upon which the AnadigmDesigner2 software is based
- These are circuit building blocks abstracted to a functional level which enable the circuit designer to design at a block level rather than worry about circuit implementation details
- These are intelligent, self contained software components that contain net lists, graphics, dialog boxes and dynamic “C” code.
Configurable Analog Modules (CAMs)

- Configurable Analog Modules (CAMs)
  - Gain Stages
  - Summing Stages
  - Rectifiers
  - Filters
  - Many others
- Each CAM is adjustable
  - Dynamic user interface – options and limits can change
CAM Dialog Window

Clock and name

Radio buttons

Parameter entry area
- **AnadigmDesigner®2**
  - Built-in signal generator, oscilloscope
  - Built-in, discrete-time functional simulator
  - Difference equation based
  - Import/export data from/to other simulation tools and programs
AnadigmDesigner®2

- Extensive help documentation
Using the AnadigmFilter™

This is the pole zero pole. You can drag pole and zero icon’s within this plot to modify the filter.

This is the frequency domain plot. You can drag all of the straight lines within this plot to modify the filter.

Choose Frequency response
Phase
Group delay
Or all tree. You can also change the graph scales here.

Choose Frequency type and approximation as a starting point.

Type filter parameters in here if you know them.

These boxes set-up the preferences and control the communication to AnadigmDesigner2.

This area tells you which CAMs and parameters will be used.

This icon builds the filter in AnadigmDesigner2, see next page.
Now you can adjust the filter and/or the CAM parameters within AnadigmDesigner2.

And you can upload changes from AD2 to AnadigmFilter using this icon.

Never change both view at the same time; always upload or download after each change.
Using the AnadigmFilter™

Complex high order single function filters are easy to make by pointing and click the standard approximations provided. To build a more complex filter eg a bandpass with two peaks.

Try these steps.

1) Build an 8 pole low pass Butterworth filter
2) Download it too AnadigmDesigner2
3) Within AnadigmDesigner2 change the 4 biquad CAM from low pass to band pass.
4) change two of the Biquad centre frequencies to $F_c=a$
5) Change the other two Biquad centre frequencies to $F_c=b$
6) Upload the circuit to Anadigmfilter

Now you see the filter response for your customer filter, using this technique almost any filter can be built.
Generating Configuration Records for the AnadigmApex devices
Generating Load Records for the AnadigmApex devices

- The AnadigmDesigner2 software can generate two general types of configuration information for the AnadigmApex devices:
  - Files in various standard forms such as Motorola s1, s2, binary, etc.
  - In the form of code and tables in the dynamic “C” code the AnadigmDesign2 generates.
Dynamic Reconfiguration – C-Code

```
/*
 * Description
 * This function sets the 3dB corner frequency of the filter in this Output cell.
 *
 * Instance Name  sCAM    sChip
 *------------------------
 * OutputCell2  _n_chip1_OutputCell2  _n_chip1
 */

/*
 * Comparator.cam
 *
 */

/*@*******************************************************************************/
@*******************************************************************************/
@
@

/*@*******************************************************************************/

/*
 * Description
 * This function controls the output polarity of the Comparator.
 *
 * Instance Name  sCAM    sChip
 *------------------------
 * Comparator1  _n_chip1_Comparator1  _n_chip1
 */

/*
 * RectifierB alf.cam
 *
 */

/*@*******************************************************************************/
@*******************************************************************************/
@
@

/*@*******************************************************************************/

Gain

@
@

/*@*******************************************************************************/

Gain

@
@

/*@*******************************************************************************/

Gain

@
@

/*@*******************************************************************************/

Gain

@
@

/*@*******************************************************************************/

Gain

@
@

/*@*******************************************************************************/

Gain

@
@

/*@*******************************************************************************/

Gain

@
@

/*@*******************************************************************************/

Gain

@
@

/*@*******************************************************************************/

Gain

@
@

/*@*******************************************************************************/

Gain

@
@

/*@*******************************************************************************/

Gain

@
@

/*@*******************************************************************************/

Gain

@
@

/*@*******************************************************************************/

Gain

@
@

/*@*******************************************************************************/

Gain

@
@

/*@*******************************************************************************/

Gain

@
@

/*@*******************************************************************************/

Gain

@
@

/*@*******************************************************************************/

Gain

@
@

/*@*******************************************************************************/

Gain

@
@

/*@*******************************************************************************/

Gain

@
@

/*@*******************************************************************************/

Gain

@
@

/*@*******************************************************************************/

Gain

@
@

/*@*******************************************************************************/

Gain

@
@

/*@*******************************************************************************/

Gain

@
@

/*@*******************************************************************************/

Gain

@
@

/*@*******************************************************************************/

Gain

@
@

/*@*******************************************************************************/

Gain

@
@

/*@*******************************************************************************/

Gain

@
@

/*@*******************************************************************************/

Gain

@
@

/*@*******************************************************************************/

Gain

@
@

/*@*******************************************************************************/

Gain

@
@

/*@*******************************************************************************/

Gain
Exercise 1: Build a Circuit that Filters and Rectifies a Signal

- Circuit will take input and both filter and rectify the signal
- This is a basic circuit that will determine
  - The frequency components of a square wave
  - The average DC value of the extracted component
Exercise 1: Set up a Biquadratic Band Pass Filter

Open AnadigmDesigner2. Click on the green box to pop up the CAM selection window.
Exercise 1: Place Biquadratic Filter in Chip Work Space

Choose (double Click) from the list of CAMs a “FilterBiquad” Filter. (this is a two pole pole Filter) and set the dialog parameters as shown on the next page.
Exercise 1: Set Biquadratic CAM parameters

Step 1: set the Clock to 2000 kHz by moving the spinners.

Step 2: Select Band pass filter by clicking on the band pass radio button.

Step 3: Set Corner Frequency to 5 kHz, gain to 1.0 and Q to 15.

Step 4: Click on OK to accept parameters.
Exercise 1: Place a “RectifierFilter” CAM in the Work Space

Choose (double Click) from the list of CAMs a “RectifierFilter” CAM. This is a rectifier with a combined single pole low pass filter.

The combination of the filter and rectifier uses the same number of op amps as a rectifier alone eliminating the need for a separate filter!

Please set parameters as shown on the next page.
Exercise 1: Set parameters for the “RectifierFilter”

Step 1: Set the Clock to 2000 kHz by moving the spinners.

Step 2: Choose a non-inverting Full wave rectifier. Note the other choices for future reference.

Step 3: Set Corner frequency to 100 KHz and gain to 1. Why 100 KHz? It lets through almost all the energy from the square wave, but blocks higher harmonics that may alias back.

Step 4: Click on OK to accept parameters.
Exercise 1: Add a Signal Generator

Set IO2 to Input and Bypass mode. Add a signal Generator by clicking on the sine wave on the tool bar and then place the signal generator here.

Next....

Double click on the signal generator to pop up the configuration window on the next page.
Exercise 1: Configure signal generator

And set Parameters as shown to create a 5Khz square wave

Next....wire the circuit up
**Exercise 1: Wire up the circuit**

Wire it up!

1) Just drag the mouse over to the contact or wire that you want to wire up

2) A wire tool graphic will appear and the contact will highlight

3) Next, click the left mouse button

Next..... Place oscilloscope probes
Exercise 1: Place Oscilloscope Probes

Place Probes

1) Depress the probe tool icon on the tool bar (to the right of the sine wave)

2) Drag it near a contact and when the contact high lights, depress the left mouse button

3) Maintain the probe colors as shown in the picture so you can follow along

Next.....

Configure the Simulator
Exercise 1: Configure the Simulator

Configure the simulator
- Click on the ‘Simulate’ tab on the tool bar and then choose the selection ‘Setup Simulation’
- Configure the options as shown

Next.....

Run the simulator
Exercise 1: Run the Simulator

Depress the “Sim” target on the screen to run. It took 110 seconds on a 2.3 GHz Pentium.

Set parameters as shown (circled in red) by adjusting the spinners or scroll bar (in Green).

Fundamental extracted from square wave.

Rectified Fundamental.

Input Square Wave 5Khz.

Note: If you were not able to build the circuit, just load “class II circuit 1.ad2” from the class package you received.
Exercise 1: Next Steps

- Change to a **bandstop** filter to block rather than pass the fundamental
- Simulate to see what a square wave with its first harmonic blocked looks like!
- Obtain average DC value of the rectified signal
- Create "C" code
- Create a circuit file in S2 format that can be loaded into a EPROM
Exercise 1: Set Biquadratic CAM parameters

Step 1: set the Clock to 2000 kHz by moving the spinners (it should already be like this)

Step 2: Select Band stop filter by clicking on the band pass radio button. This will block rather than pass the fundamental 5KHz signal.

Step 3: Set Corner Frequency to 5 KHz, DC gain to 1.0 and Q to 15 and HF Gain to 1

Step 4: Click on OK to accept parameters
Exercise 1: Run the Simulator Again

Depress the “Sim” target on the screen to run. It took 110 seconds on a 2.3 GHz Pentium.

Set parameters as shown (circled in red) by adjusting the spinners (in Green):

- Square wave less the fundamental
- Rectified

Note: If you were not able to build the circuit, just load “class II circuit 2.ad2” from the class package you received.
Exercise 1: Determine Average DC value of rectified signal

Load the circuit “class II circuit 2.ad2” which is included with your class email package. Note that a Biquadratic, non-inverting low pass filter is added to extract the average DC value of the rectified signal.

See the simulation trace on the next page. Run the simulation as before by depressing the “Sim” tool bar target as before.
Exercise 1: Determine Average DC value of rectified signal

Average DC value, in yellow.
Exercise 2: Create “C” code

This exercise will show you how to create “C” code which will enable you to dynamically control the AnadigmVortex device from a processor (uP, DSP, PIC, etc).

Choose the “Dynamic Config” tab from the menu and then choose “algorithmic method” and then the panel to the left will pop up.
Exercise 2: Create “C” code

Step 1: Specify the file names.

Step 2: Specify the destination directory.

Step 3: Click on the Generate button to create C code.

Step 3: This message box will be displayed when complete. Open the “C” code files and take a look.
Exercise 2: Create “C” code

Here are the files on the desk top
Open them with WordPad (not notepad!) and take a look!
Exercise 2: Create “C” code

Here an excerpt of code that will manipulate parameters for a band pass filter.

If you were not able to do this, the files are included in the class package.

```c
void an_SetBQBandStopFilterII(an_CAM nCAM, double Fo, double Glow, double Ghigh, double Q)
{
    static const double an_Pi = 3.1415926535897931;
    static const double an_TwoPi = 6.2831853071795862;
    static const int maxCA = 255;
    static const int maxCB = 510;

    int dCI, dCpp, dC2, dC3, dC4, dCA, dCB;
    int CA1, CB1, CB2;
    int Q1=0, Qpp=0, Q2=0, Q3=0, Q4=0, QA=0, QB=0; // final high Q biquad cap values
    double aFoA, aFoA2, aQ, aGlo, aGhi; // achieved parameter values

    long clocka = an_GetClockFrequency(nCAM, an_CAMClock_ClockA);
    double FCdiv2PI = clocka/ an_TwoPi;

    // Limits in this module are extremely complicated due to the interrelation of parameters
    // Calculate limits and clamp any desired values as required
```
Exercise 3: Create a Configuration File

Choose the “Configure” tab and then “Write configuration to a file…”
Exercise 3: Create a Configuration File

Select the file format you wish to save in and then click on save and the dialog box on the following page will pop up.
Exercise 3: Create a Configuration File

Specify address and hit OK
Exercise 3: Create a Configuration File

Open the file with note pad
Exercise 3: Create a Configuration File

And it looks like this
Exercise 4: Build a filter with AnadigmFilter

Open Anadigm Filter with a blank workspace and select Anadigm filter

Next…
**Exercise 4: Build a filter with AnadigmFilter**

Select High Pass and then click on download icon to synthesize your filter!!!
Exercise 4: Build a filter with AnadigmFilter

And here it is! AnadigmFilter built a 5 pole Butterworth high pass filter for you!

Try different types of filters and AnadigmFilter will rebuild the filter for you!
Thank You For Attending Anadigm Training!
Next Steps: Start working with Anadigm Programmable Analog

- Purchase Development kit for $199.00
- Our distributor www.NuHorizons.com
- Go to the internet at www.anadigm.com for more product information
- You are ready to start working with the Anadigm development kit.