TI | Developer Conference
March 7-9, 2007 • Dallas, TX

Minds in Motion

Using Hybrid Programming in DSP Lab Courses

N. Kehtarnavaz and V. Peddigari

UTD

Signal and Image Processing Lab
University of Texas at Dallas

TIDC, March 07

Outline

- Motivation
- What is Hybrid Programming
- What is Graphical Programming
- Pros & Cons of Textual and Graphical Programming
- Advantages of Hybrid Programming
- Examples of DSP Systems Coded in Hybrid Programming
  - Digital Filtering on TI DSK Platform
  - Cochlear Implant System
  - Overexposure Correction System
- Summary
- Live Demo

Motivation

- Observation – Often, in DSP lab courses, students (in particular undergraduates) spend a fair amount of their time debugging text-based codes. Leaving not much time for analyzing designed DSP systems.
- To avoid this problem, the use of hybrid programming in DSP lab courses is recommended in order to achieve DSP system building in a more time efficient manner.

What is Hybrid Programming?

- Hybrid programming means performing a combination of textual (e.g., MATLAB®) and graphical (e.g., LabVIEW®) programming.

- Hybrid programming allows one to bring together the preferred features of textual and graphical programming.

Ways to Achieve Hybrid Programming

- Hybrid programming can be achieved in a number of different ways, e.g.:
  - Using a C / MATLAB Dynamic Link Library (DLL) node within the LabVIEW graphical programming environment.
  - Using MATLAB in the SIMULINK® programming environment.
  - Using MathScript feature of LabVIEW 8.0 or higher within the LabVIEW programming environment.

What is Graphical Programming?

- Unlike text-based programming, it offers block-based or more intuitive approach to code development.
- **LabVIEW** and **SIMULINK**: Most widely used graphical programming environments.

- A comparative study between LabVIEW and SIMULINK was conducted by undergraduate students in a DSP lab course at UTD. Outcome reported at ICASSP06 Education Session.

Overview of Comparative Study

- On average, students took 2 to 4 hours to finish each lab (7 labs total), depending upon the complexity of the design problem.
- Level of prior exposure to the environments: None – learned them in the course.
- Students were asked to rate LabVIEW and SIMULINK for each of the labs with respect to a specified set of 8 criteria.
- Students rated each criterion on a scale of 0 to 10 with 10 representing the highest rating.

Evaluation Criteria (1)

- **Learning curve:** Ease of getting familiar with the programming environment.
- **Ease of use**: Environment offering easy to use features for code development, reuse, and expansion.
- **Programming constructs**: Having sufficient programming constructs and data structures.
- **Breadth of functionality:** Rich set of “plug-n-play” building blocks and components available for DSP system design.

Evaluation Criteria (2)

- **Graphical User Interface (GUI):** GUI capabilities of the programming environment; ease of interaction with the system at run-time.
- **Debugging features**: Easy to debug programs; availability of graphical debugging tools.
- **DSP Test Integration tools**: Ease of interacting/integrating the environment with other software (TI Code Composer Studio™ IDE) and hardware platforms (TI TMS320C6416 and TMS320C6713 DSK).
- **Help resources**: Richness of technical documentations and online help; easy to understand documentations.

Comparative Study Outcome

- In general, students found LabVIEW and SIMULINK to possess similar features. More or less, equally rated by the students.

- However, one place where students clearly preferred using LabVIEW over SIMULINK was the GUI and interactive capabilities of LabVIEW.

LabVIEW Overview (1)

- LabVIEW
  - A graphical programming environment developed by National Instruments (NI) which allows one to design systems without needing to have any prior text-based programming experience.

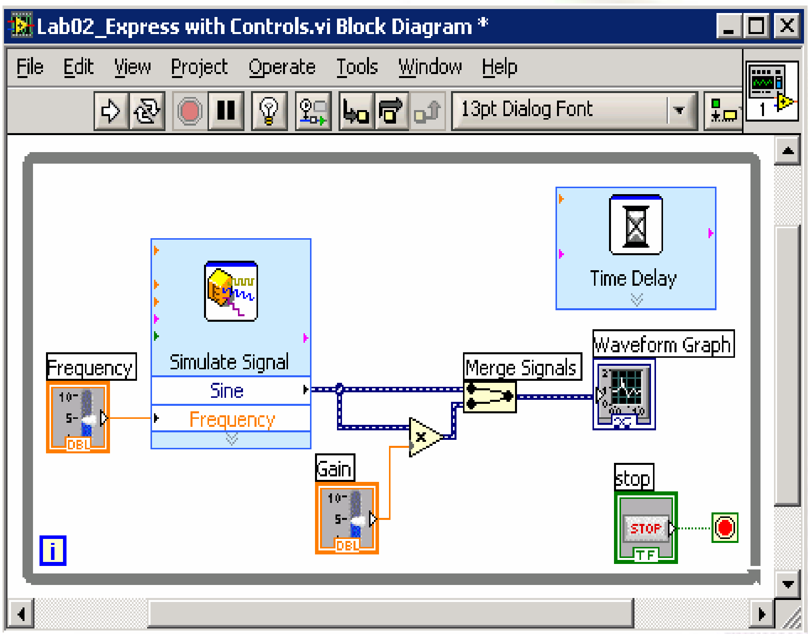
- Programs in LabVIEW
  - Virtual Instruments (VI): Graphical modules that are put together in an intuitive flowchart-like manner.
  - A system design is achieved by integrating different blocks or subsystems.

LabVIEW Overview (2)

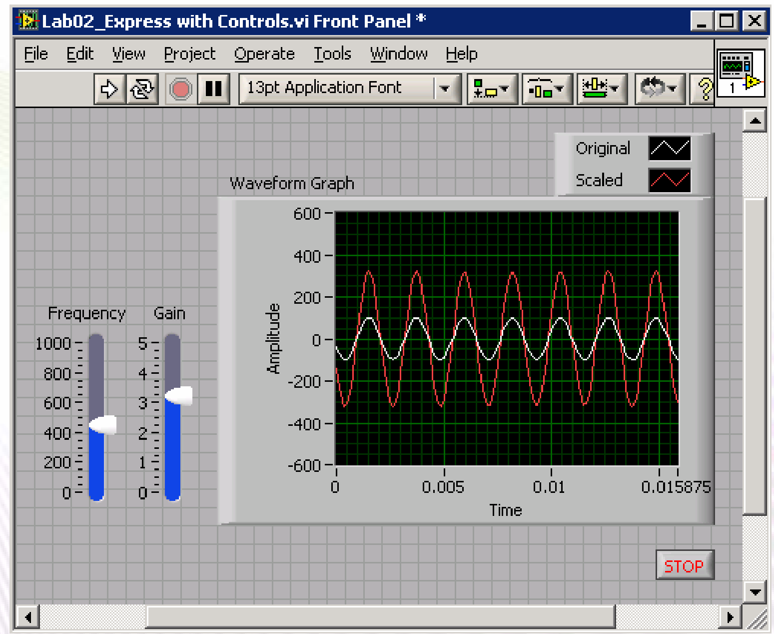
- A VI consists of two parts:
  - Front Panel
    - The interactive graphical user interface incorporating various controls and displays.
  - Block Diagram
    - The interconnected building blocks of a function or system similar to a flowchart.

LabVIEW Overview (3)

- Sample LabVIEW program (signal generation and amplification)



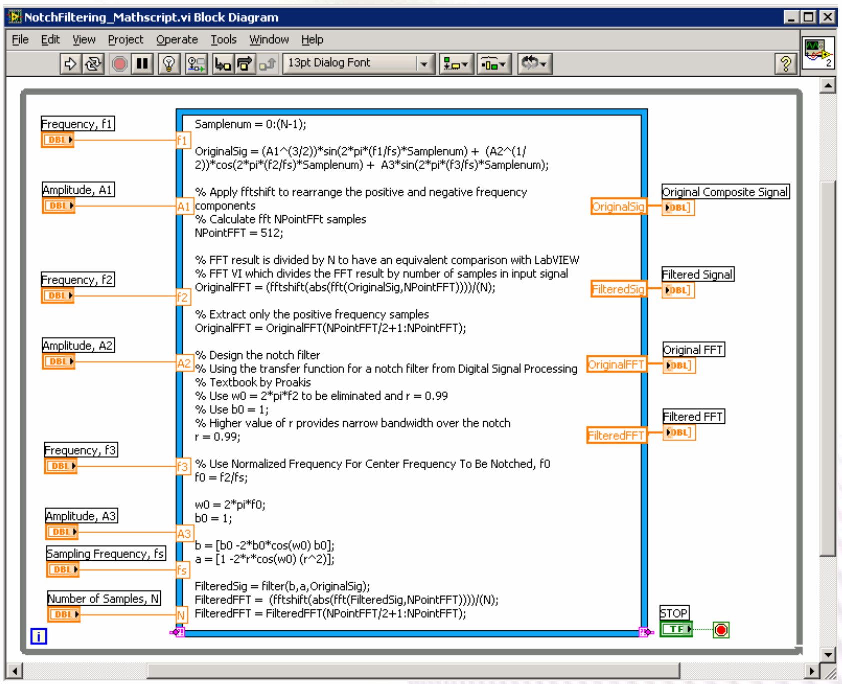
Block Diagram



Front Panel

Pros of Graphical Programming Notch Filtering Example (1)

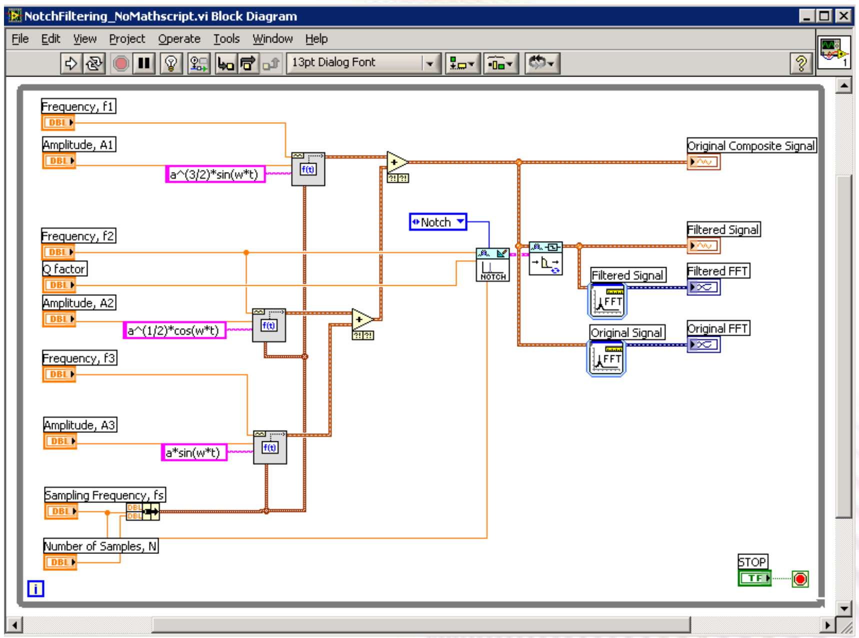
- If done in textual code, the code will not be compact and require the user to know about various parameters of the functions.



Textual Code

Notch Filtering Example (2)

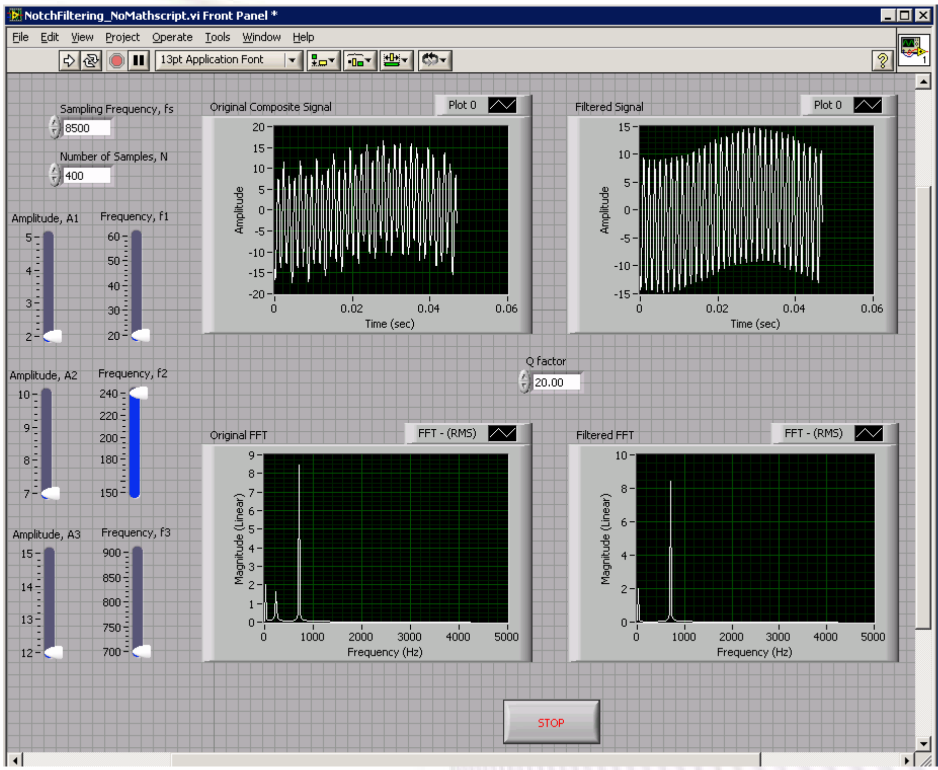
- Graphical code is preferred here since
  - Reusable off-the-shelf functional blocks are available.
  - Simple to set parameters in an interactive way.



Graphical Code

Notch Filtering Example (3)

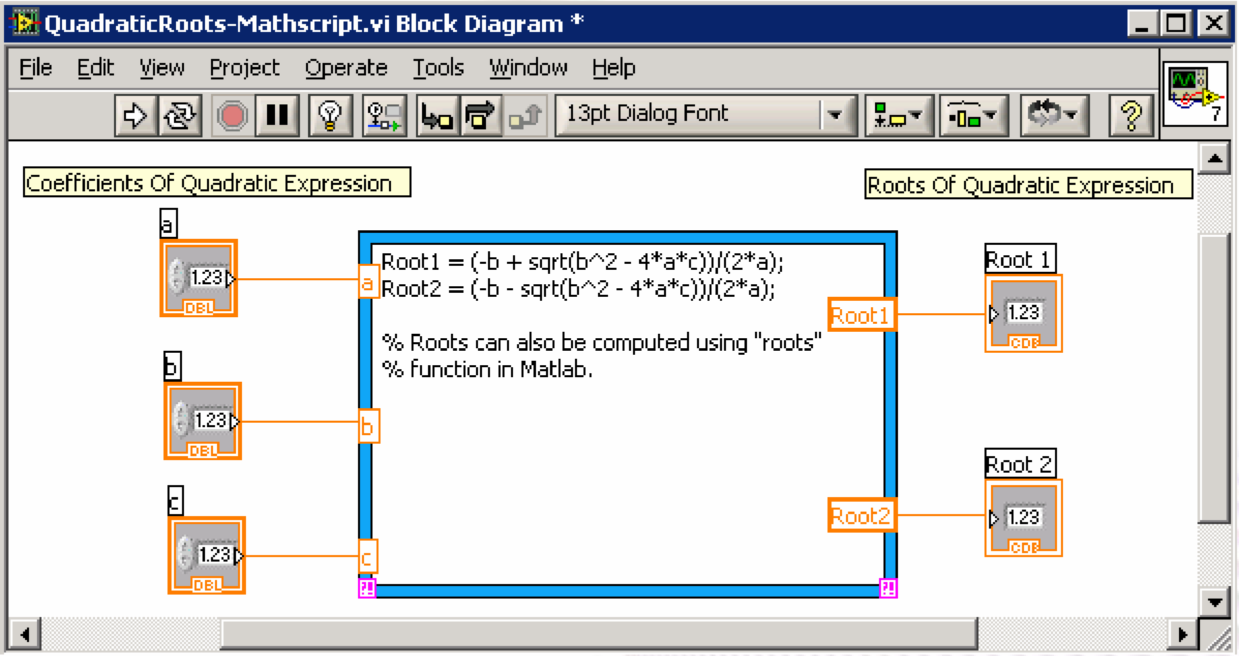
- Interactive GUI capability.



Front Panel

Cons of Graphical Programming Quadratic Roots Example (1)

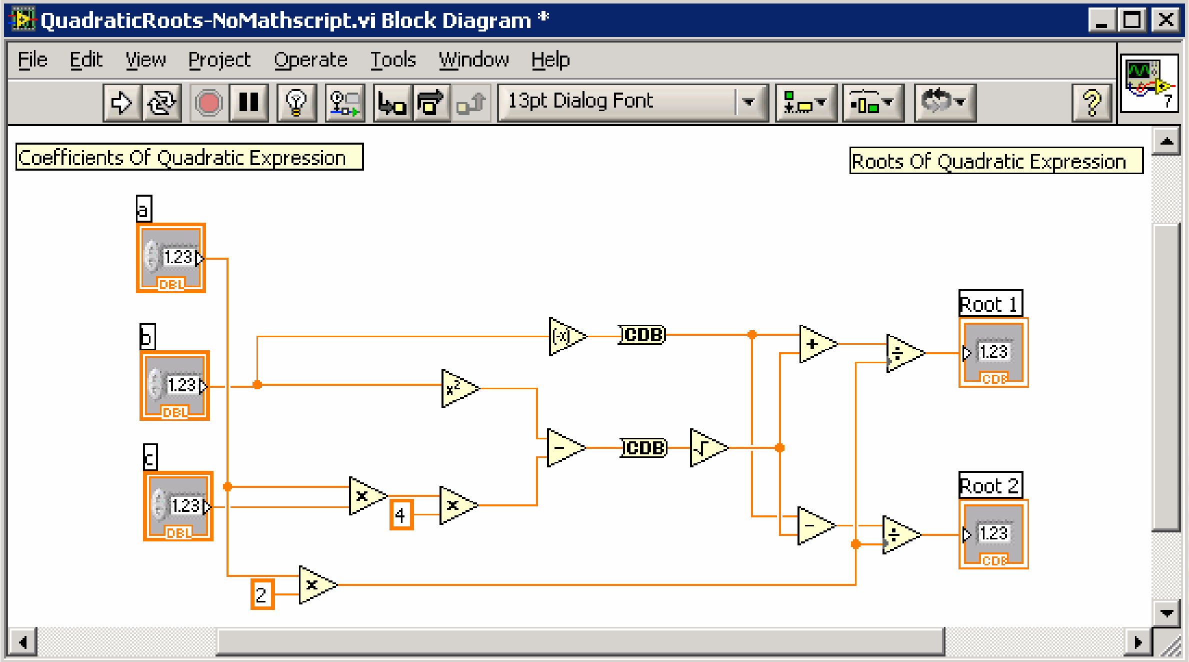
- Textual code is preferred here since it provides compact code size & is easier to make modifications.



Textual Code

Quadratic Roots Example (2)

- The use of graphical code here is cumbersome and makes the code difficult to follow.



Graphical Code

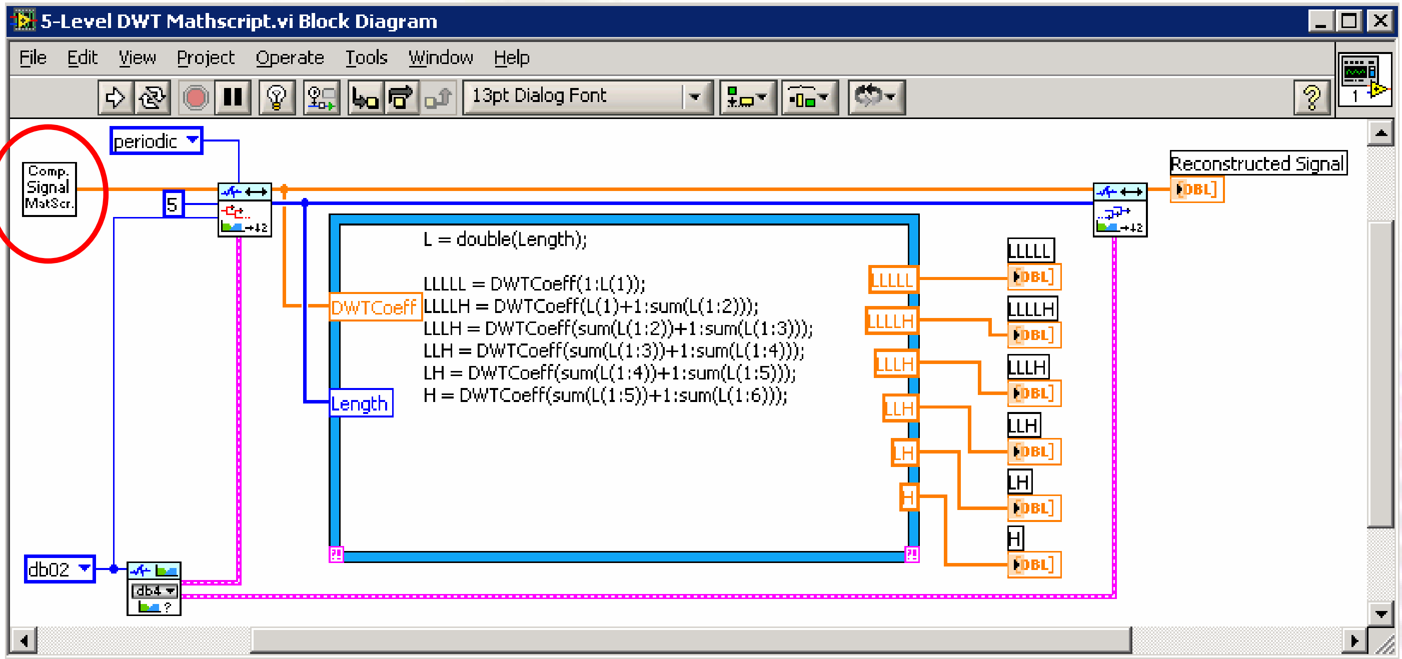
Combine Textual and Graphical - Hybrid Programming

- Hybrid programming blends the advantages of both graphical and textual environments:
  - Builds upon the prior text-based (C, MATLAB) programming experience of students.
  - Provides an intuitive or block-based approach towards designing DSP systems.
  - Allows a modular and hierarchical system design.
  - Furnishes an interactive GUI.

Hybrid Programming Example - DWT (1)

- Hybrid programming blends the advantages of textual and graphical programming.

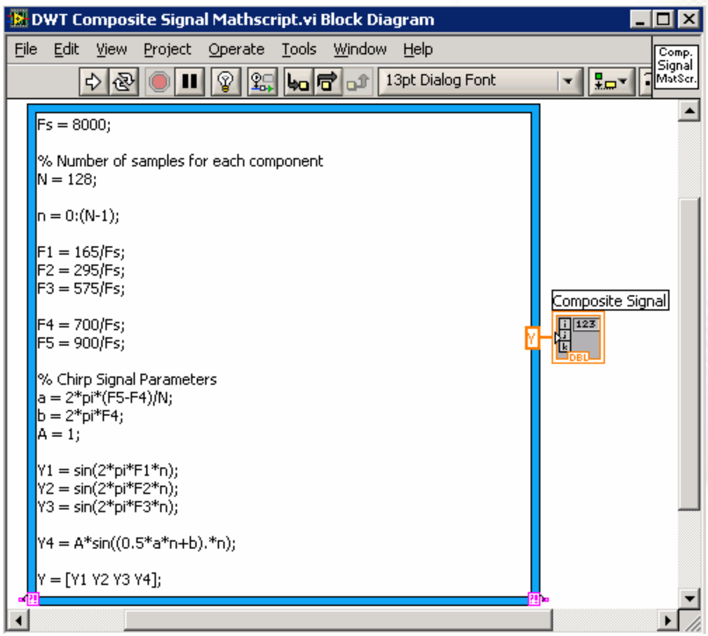
Graphical programming preferred here



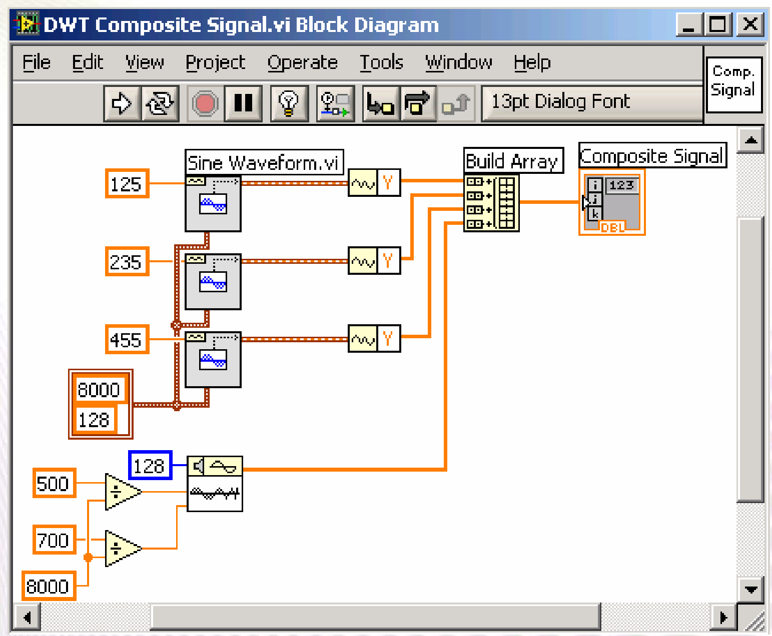
Hybrid (Textual + Graphical) Code

Hybrid Programming Example - DWT (2)

- Signal generation part using graphical programming is easier and more time efficient to do due to availability of built-in functional blocks.
- DWT computation part using textual is easier and more time efficient to do.
- Hybrid code is modular by using blocks or sub-systems.



Textual

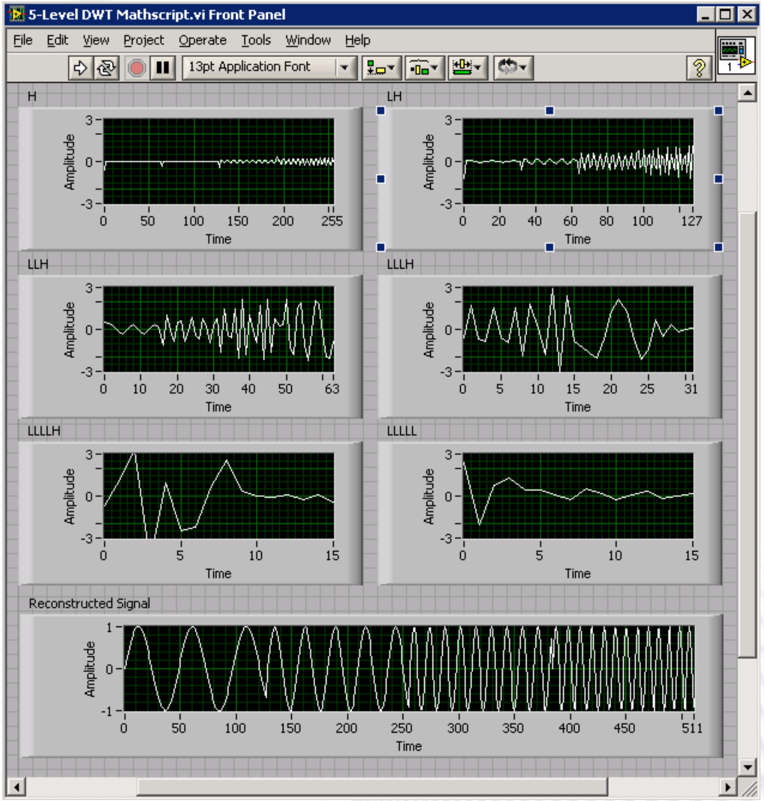


Graphical

Minds in Motion

Hybrid Programming Example - DWT (3)

- Provides an interactive display.



5-Level DWT Front Panel

Example DSP Systems Built Using Hybrid Programming

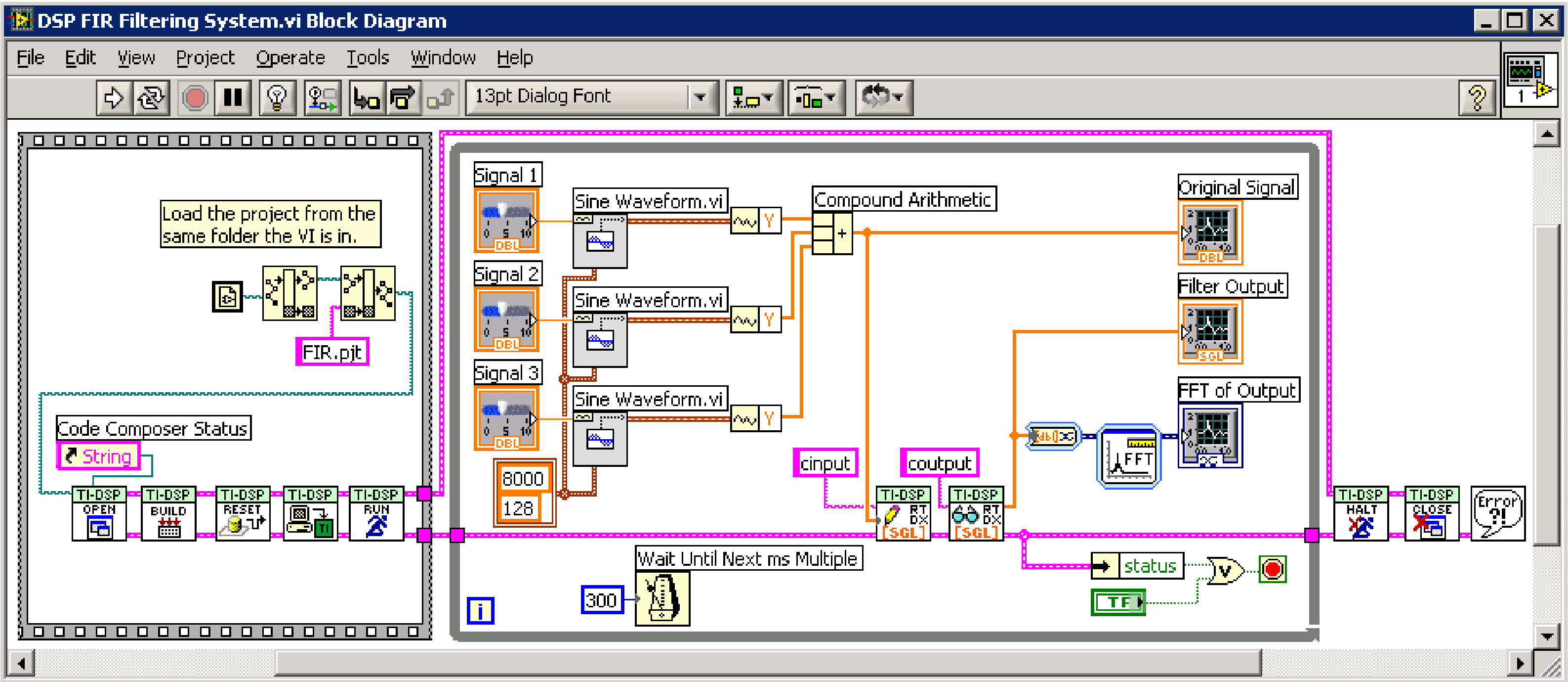
- Sample DSP systems built in DSP lab courses at UTD:
  - Digital Filtering on TI DSK platform.
  - Real-Time Simulation of Cochlear Implant System on Personal Computer (PC) and Personal Digital Assistant (PDA).
  - Cell-phone Camera Overexposure Correction.

Digital Filtering on TI DSK Platform (1)

- Filtering system on PC side, actual filtering operation written in C performed on TI TMS320C6x™ DSK platform.
  - – Communication between LabVIEW and DSK board done via RTDX™ (Real-Time Data Exchange).

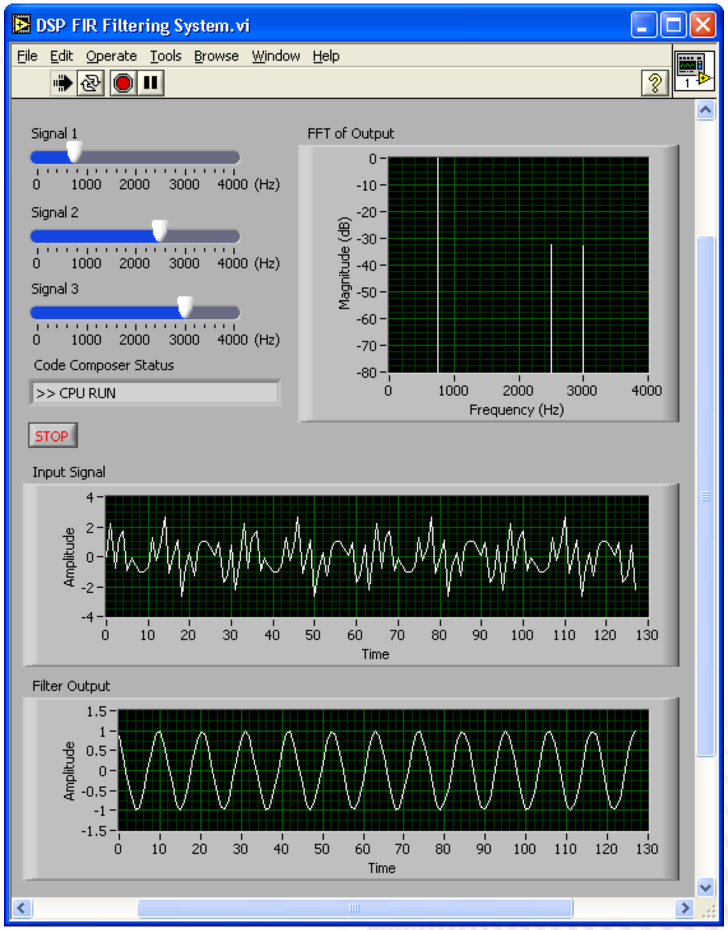
Digital Filtering on TI DSK Platform (2)

- The filtering operation part written in C runs on the TI DSP processor via RTDX.



Hybrid Code for Digital Filtering on TI DSK Board

Digital Filtering on TI DSK Platform (3)



Digital Filtering Front Panel

Cochlear Implant System (1)

- Cochlear implants are used to restore partial hearing in profoundly deaf people.

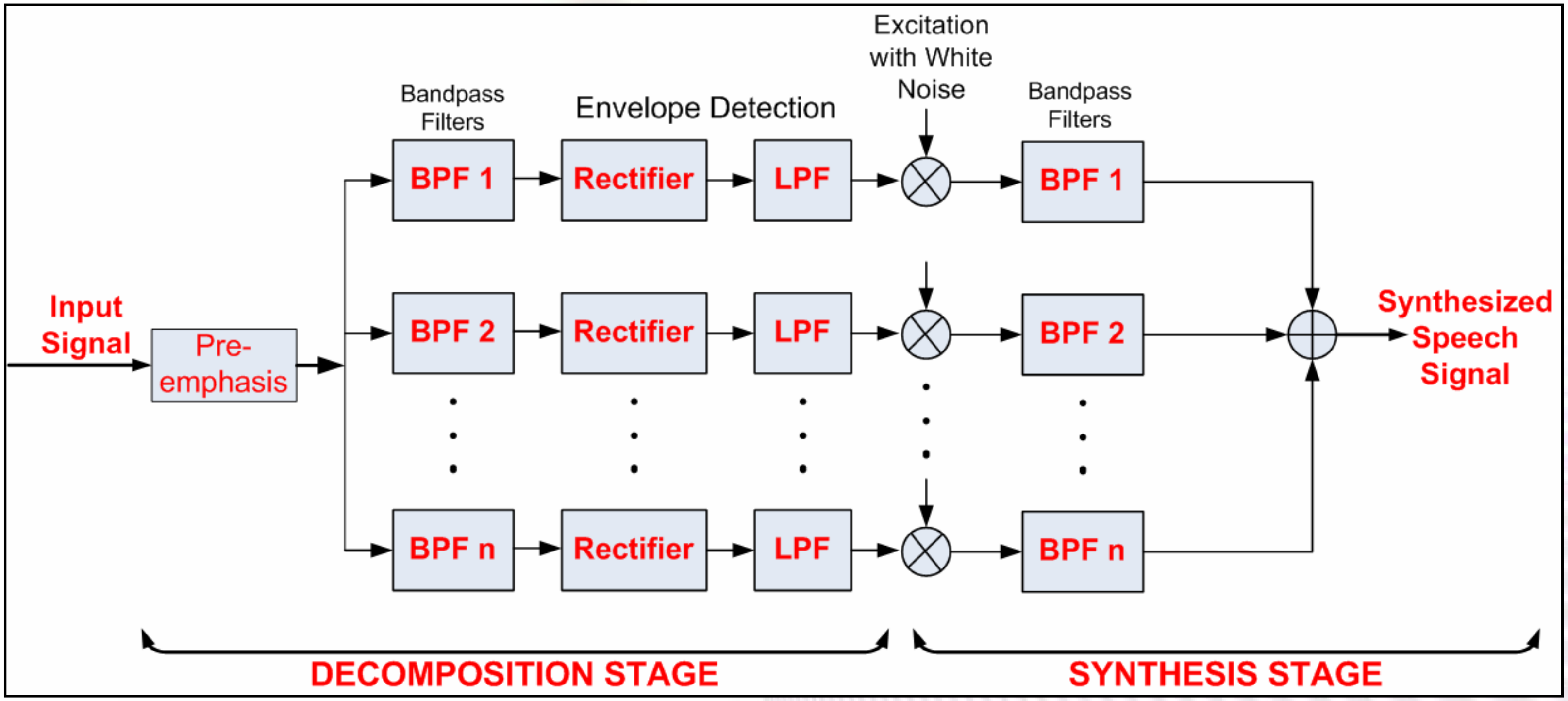
- A cochlear implant consists of three components:
  - A microphone that picks up the sound.
  - A signal processor that converts the sound into excitation signals.
  - A transmission system that transmits the excitation signals to the implanted electrodes.

Cochlear Implant System (2)

- Signal Processor
  - Breaks the input acoustical signal into different frequency bands or channels.
- Various signal processing strategies can be used to convert acoustic signals into excitation signals.
  - Popular Strategy: Continuous Interleaved Sampling (CIS)
- Different strategies can be used for signal synthesis.
  - Popular Strategy: Noise-band Vocoder

Cochlear Implant System (3)

- CIS signal processing



Real-Time Simulation (1)

- Hybrid Programming approach was chosen to achieve real-time simulation on
  - Personal Computer (PC)
  - Personal Digital Assistant (PDA)

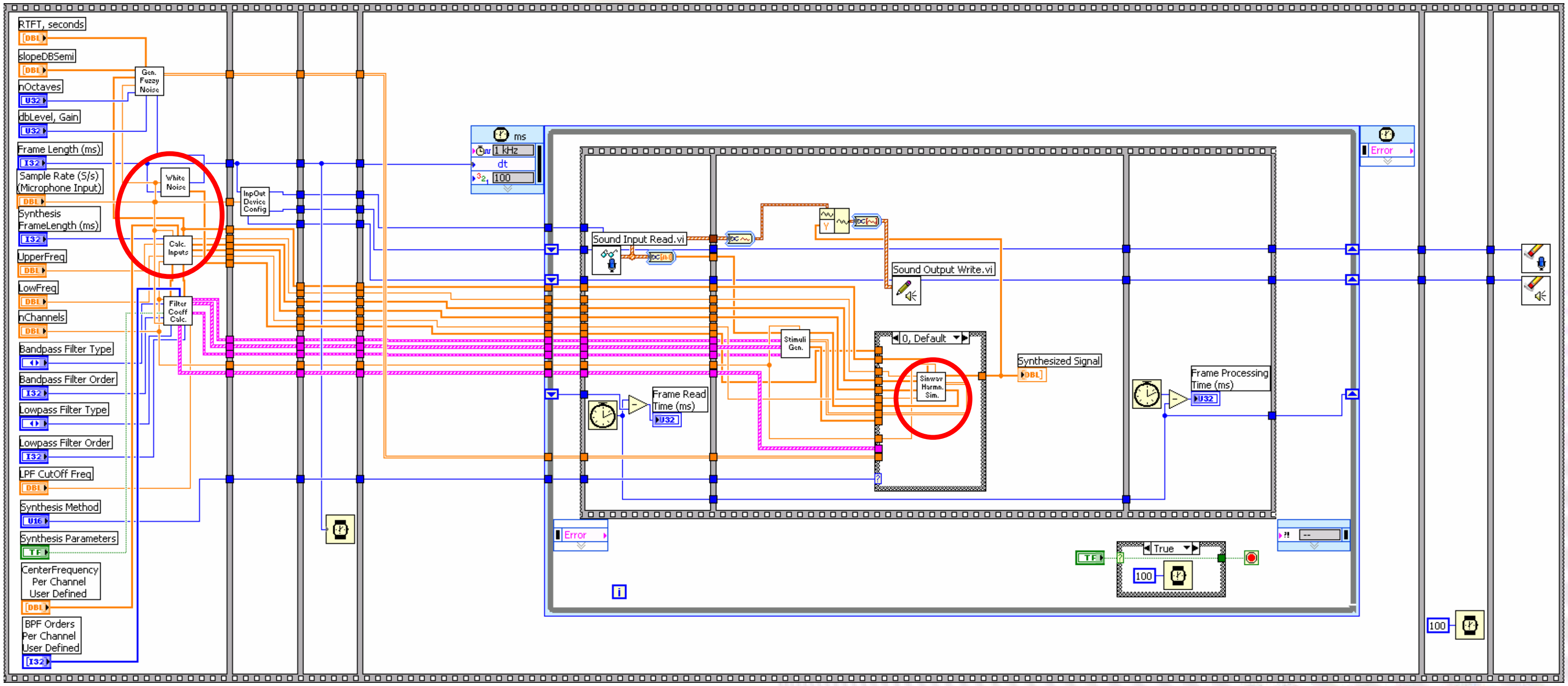
- PC Implementation
  - Filtering stages such as Band Pass Filtering (BPF), Low Pass Filtering (LPF) were written using off-the-shelf graphical blocks.
  - Rectification, Pre-emphasis & Noise excitation were implemented using textual LabVIEW MathScript or MATLAB® Script Node.
  - Acquiring sound and playback to speakers were done by utilizing the built-in graphical subsystems.

Real-Time Simulation (2)

- PDA Implementation
  - To achieve real-time throughput, optimized DLLs written in C for both decomposition and synthesis stages were used.
  - Acquisition and playback were done using optimized built-in functions of LabVIEW.
  - Other optimization steps included efficient memory allocation and performing fixed-point arithmetic.

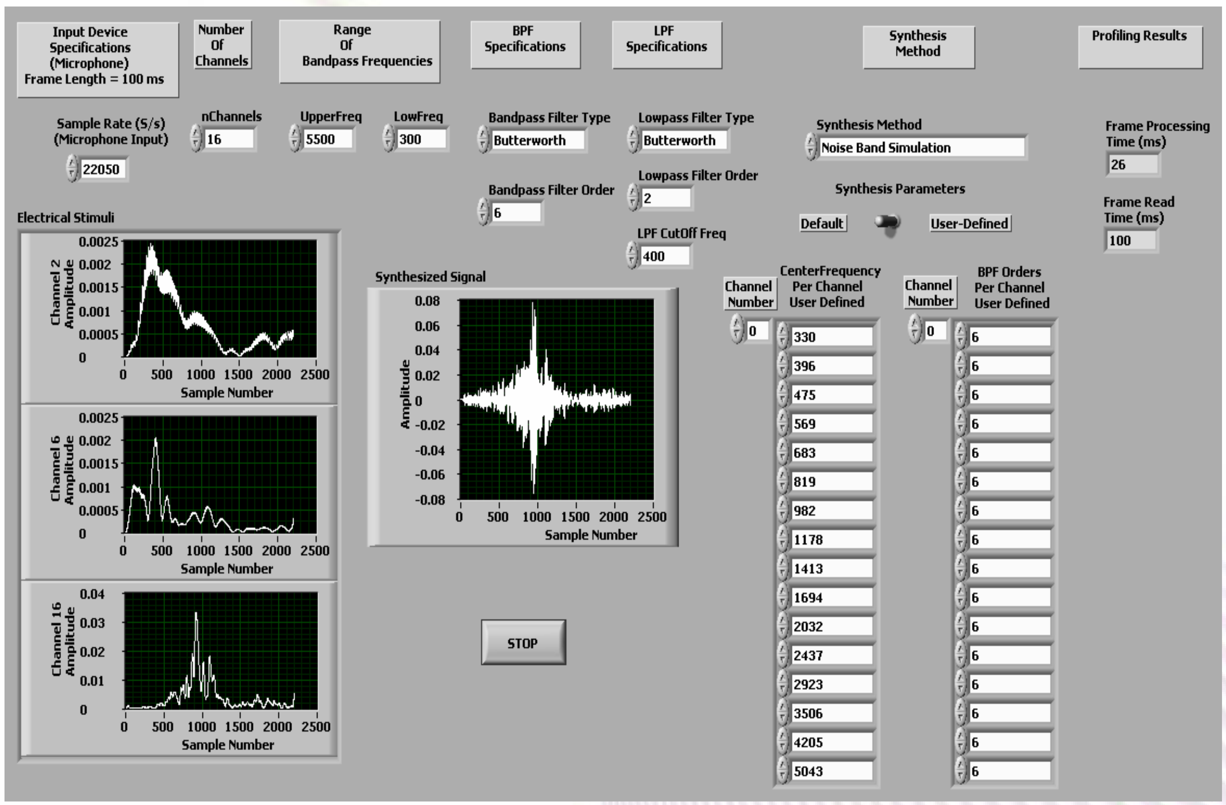
Real-Time PC Implementation (1)

- Block Diagram - Highlighted subsystems were implemented using textual programming since they involved algebraic computations.



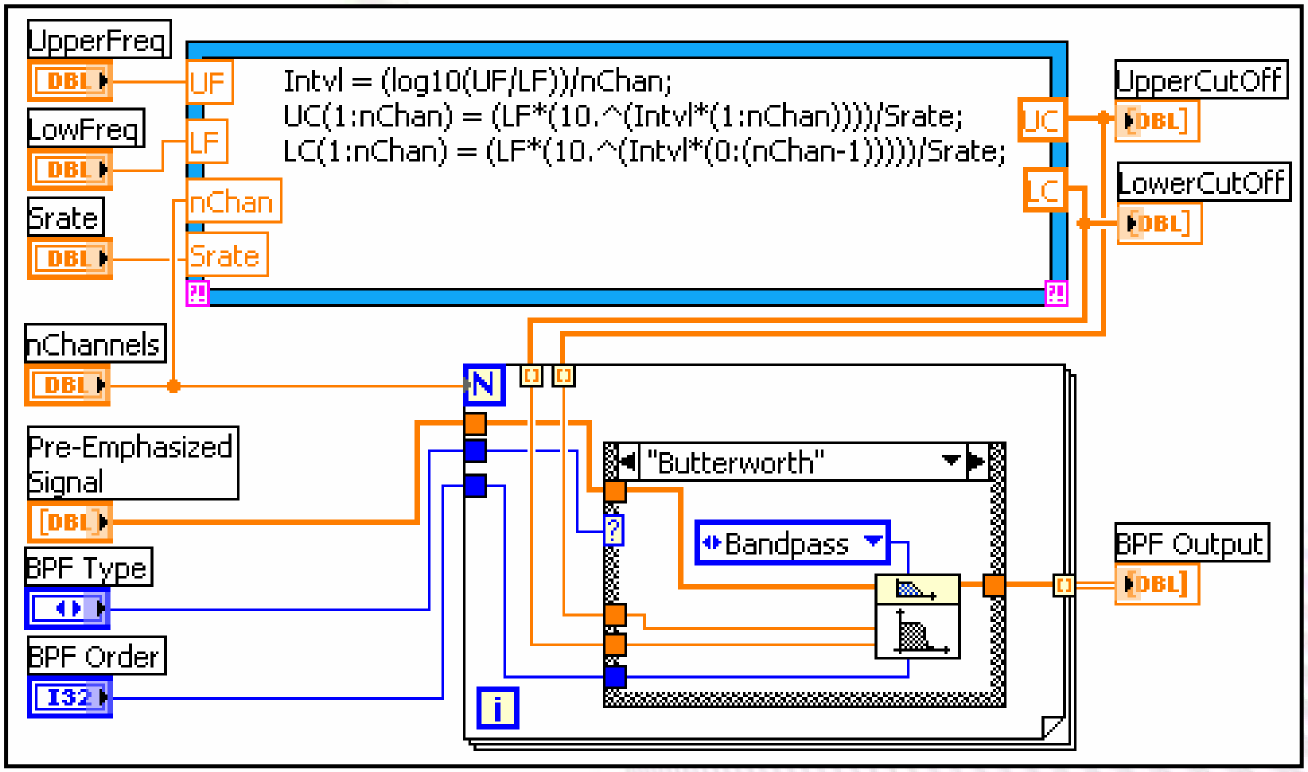
Real-Time PC Implementation (2)

- Front Panel



Band Pass Filtering Stage on PC

- Block Diagram of a sample subsystem done in hybrid programming.



Comparative Study (1)

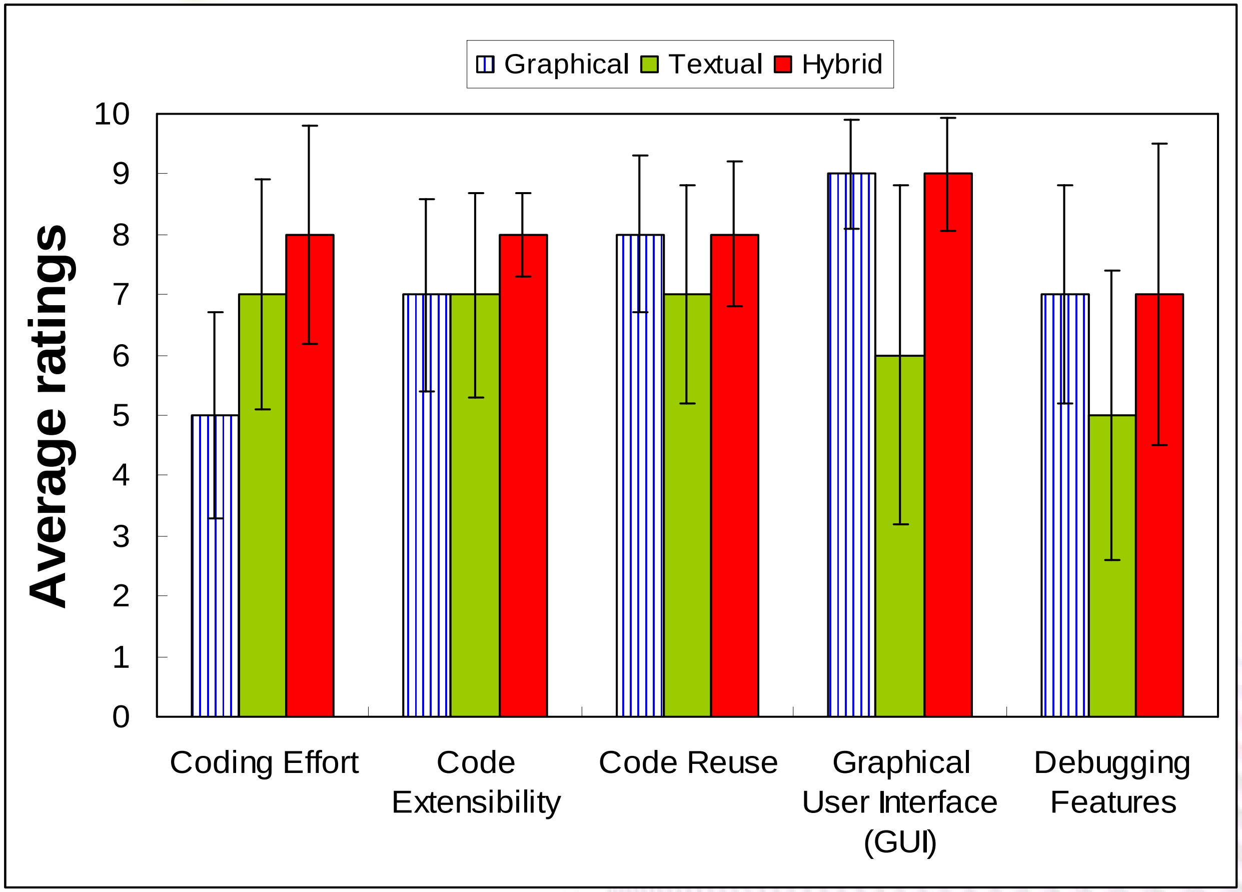
- As a DSP lab class project at UTD, students were asked to implement the CIS strategy on a PC using the three programming approaches: graphical, textual and hybrid.

- Students were asked to rate a specified set of 5 criteria on a scale of 0 to 10, with 10 representing the highest rating or score.

Comparative Study (2)

- The following criteria were used to compare the three programming approaches:
  - Coding Effort
    - Amount of time spent to make codes operational
  - Code Extensibility
    - Ease of modifying or extending the existing code
  - Code Reuse
    - Ability to use off-the-shelf blocks such as DLLs for designing more complex systems
  - Graphical User Interface (GUI)
    - Interactive user controls and displays
  - Debugging Features
    - Efficient debugging tools to reduce code development time

Comparative Study (3)



Comparative Study Outcome

- Hybrid programming approach was ranked higher when compared to textual programming over all the criteria.
- Hybrid programming was ranked higher than graphical programming for the coding effort and code extensibility criteria and was preferred equally with graphical programming for the other criteria.

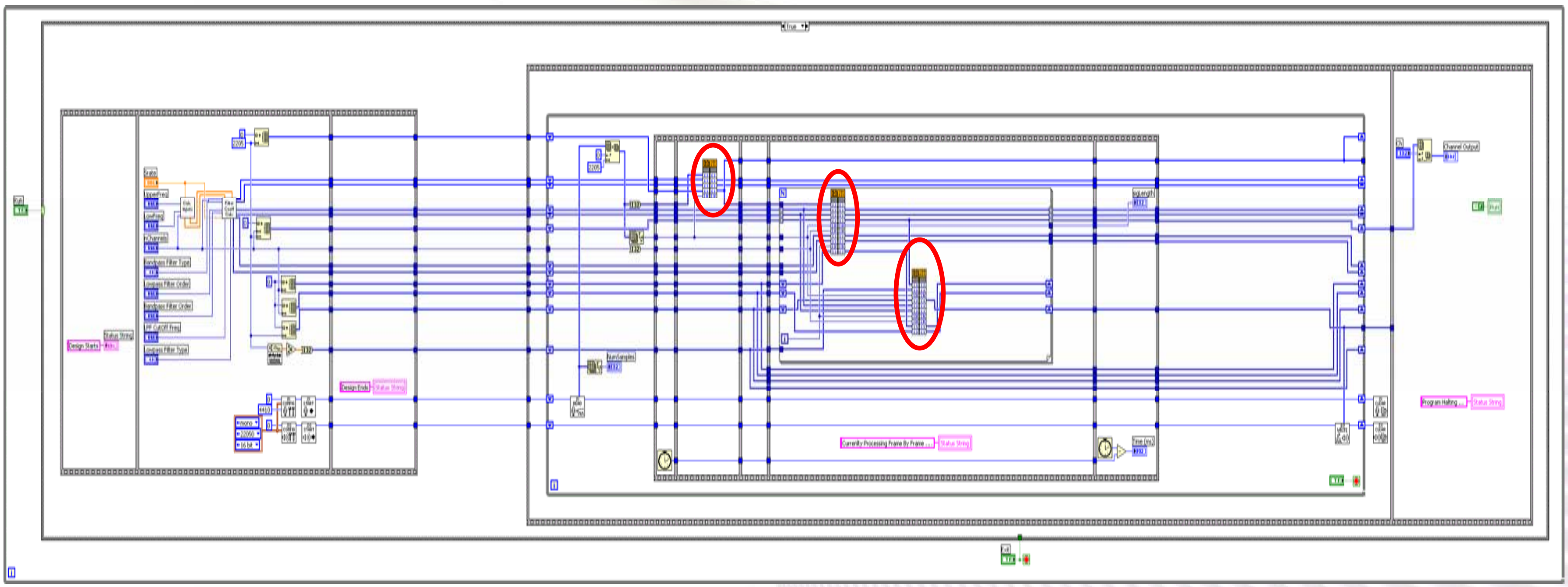
Hybrid Programming Features

- Hybrid programming benefits from useful GUI graphical features, and achieves modularity through its hierarchical approach to system design.

- Hybrid programming builds upon the prior student experience with textual programming to perform algebraic computations within a compact code size.

Real-Time PDA Implementation (1)

- Block Diagram - Highlighted subsystems denote the DLLs written in C to achieve real-time throughput on the PDA platform.



Real-Time PDA Implementation (2)

- To be presented at ICASSP07 Biomedical Applications Session.

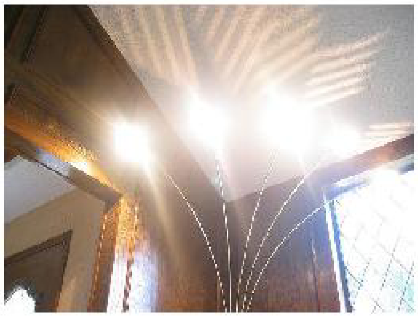
Overexposure Correction for Cell-Phone Cameras (1)

- Fusing dual-exposure images to correct for overexposed areas in the auto-exposure image captured by a cell-phone camera.

- Hybrid programming utilized:
  - SIMULINK and MATLAB were used to implement three fusing algorithms and to compare their complexity and performance.

Overexposure Correction (2)

AE Image



LE Image

Averaging Fusion

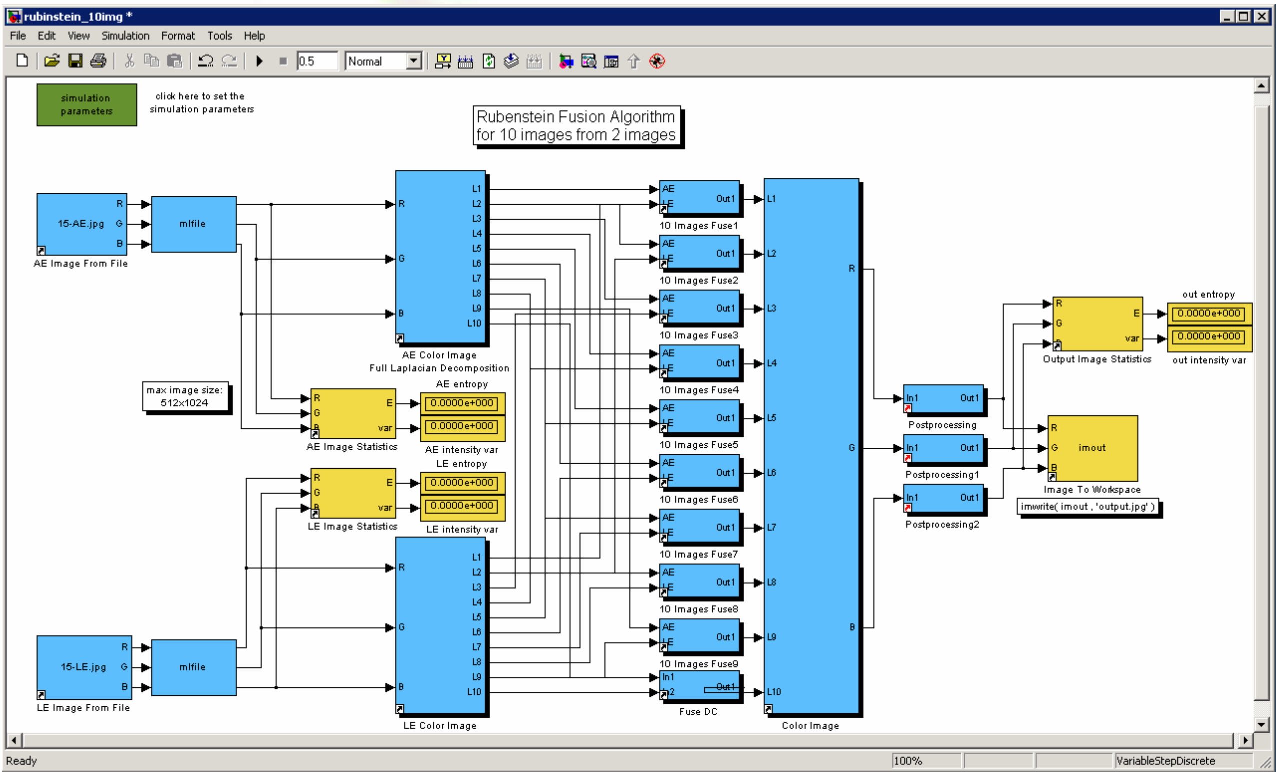
Rubenstein Fusion

Goshtasby Fusion

Our Fusion

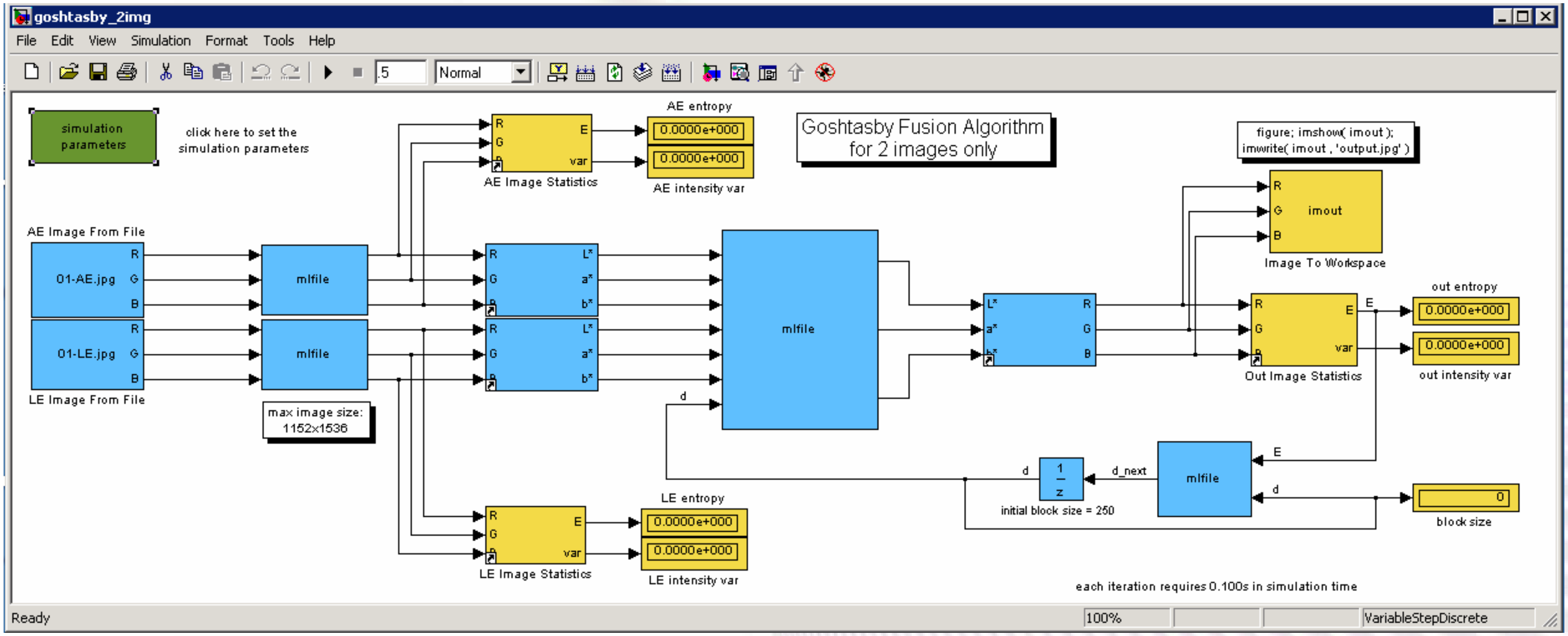
Overexposure Correction (3) – Algorithm 1

- Hybrid Programming in Simulink Environment



Overexposure Correction (4) – Algorithm 2

- Hybrid Programming in Simulink Environment



Summary

- DSP lab courses can greatly benefit from hybrid programming, in particular when students are asked to build complex DSP systems.

- Hybrid programming offers:
  - Advantages of textual and graphical programming.
  - Code flexibility and interactivity.
  - Shorter code development (system building) time.

- Teaching Materials:
  1) N. Kehtarnavaz and N. Kim, *Digital Signal Processing System-Level Design Using LabVIEW*, Elsevier, 2005.
  2) N. Kehtarnavaz, *Real-Time Digital Signal Processing Based on the TMS320C6000*, Elsevier, 2004.
  3) http://www.utdallas.edu/~kehtar/LabVIEW/

Live Demo

- Digital Filtering on TI DSK Platform

- Real-Time simulation of Cochlear Implant System on
  - Personal Computer (PC)
  - Personal Digital Assistant (PDA)

Thank You

Questions?