Design and Verify Embedded Signal Processing Systems Using MATLAB and Simulink

Giorgia Zucchelli, Application Engineer, MathWorks
10 January 2013, Technical University Eindhoven
Agenda

- Introduction to Model Based Design
- Application example: Parametric Audio Equalizer
  - Targeting a fixed-point embedded processor
- Workflow overview
- Conclusions
INTRODUCTION
What is Model Based Design?

- Methodology to design complex systems
  - Using models and simulation
  - Using tools for automation
Why using Model Based Design?

- Find errors early
- Reduce costly prototypes
- Increase productivity
1. Focus on algorithmic design
2. Anticipate implementation
3. Verification test-benches
Improve team communication with multi-domain executable specifications

- Many trusted functions
- Use the most suitable modeling approach
Achieve early verification with refined models anticipating real impairments

- Bit-true simulation
- Multi-domain physical models
Rapid prototyping with code generation: less debugging, better design

- C / C++
- Synthesizable HDL
One testbench fits all: unambiguous verification of the specs

- System-level test
- Co-simulation
- “Hardware in the loop” verification
APPLICATION EXAMPLE
Demo: Parametric Audio Equalizer
Digital filters used to adjust the frequency content of an audio signal

- Parametric response that can be run-time controlled
- Three band equalizer
  - Low Band: 60 to 1500 Hz
  - Mid Range: 1200 to 4800 Hz
  - High Range: 4800 to 12 kHz
  - Amplitude range: -8 to +8 dB
TEXAS INSTRUMENTS
DM6437 EVM
Target #1: TI DM6437 EVM
DM6437 EVM - Processor

- Highest-performance fixed-point DSP generation in the TMS320C6000™ DSP platform
- Very-long-instruction-word (VLIW) architecture developed by Texas Instruments (TI)
- Some of the specs:
  - 2.5-, 2-, 1.67, 1.51-, 1.43-ns Instruction Cycle Time
  - 400-, 500-, 600-, 660-, 700-MHz C64x+™ Clock Rate
  - Eight 32-Bit C64x+ Instructions/Cycle
  - 3200, 4000, 4800, 5280, 5600 MIPS
DM6437 – Software stack

DSP Software Architecture

- TI Embedded Processor
- Software Architecture

- Codec Engine
- DSP Link
- CCStudio IDE 3.3/4.x
- TI Codecs (audio, speech, video, imaging)
- BIOS – DSP Operating System
- Drivers
- 3P SW/Codecs
- Customer DSP Algorithms/Applications

Legend:
- Gray = Industry Standard OS S/W Component – Free
- Orange = TI Provided Components - Free
- Light Pink = Customer or 3rd Party Code
WORKFLOW OVERVIEW
PC-Based Audio Prototyping

Data source → Simulink / MATLAB → Data analysis

Simulink - Host

requirements

Design

- Algorithms
- Fixed-Point

Implementation

- C, C++
- MCU
- DSP
- Processors

Integration

Test & Verification
Fixed-Point Modeling

Data source → Fixed-Point Simulink / MATLAB → Data analysis → Simulink - Host

Requirements
- Algorithms
- Fixed-Point

Design
- C, C++
- MCU, DSP, Processors

Implementation

Integration

Test & Verification
C Code Verification

Data source → Black Box S-function → C code → Data analysis

Simulink - Host

Requirements
- Design
  - Algorithms
  - Fixed-Point

Implementation
- C, C++
  - MCU
  - DSP
  - Processors

Integration

Test & Verification
Automatic C Code Generation

Data source → Fixed-Point Simulink / MATLAB → Embedded C

Data analysis

Requirements

Design
- Algorithms
- Fixed-Point

Implementation
- C, C++
- MCU
- DSP
- Processors

Integration

Test & Verification
Processor-in-the-Loop
On-Target Rapid Prototyping

Simulink - Host

Data source

Simulink / MATLAB

Data analysis

ADC

Embedded C

DAC

Target

Requirements

Design

Implementation

Test & Verification

Algorithms

Fixed-Point

C, C++

MCU, DSP, Processors

INTEGRATION
Fixed-point design: motivation

- ASIC/FPGA or fixed-point DSP implementation
- Saves power and/or cost
- Word length and fraction length must be specified

Diagram:
- L-N
- N
- L

- sign + integer
- fractional
Fixed-point design: challenges

- It introduces degradation that must be assessed
  - Quantization error
  - Overflow / Underflow

- All variables must be converted including internals: this is error prone and often uninspiring
Fixed-point data type in MATLAB / Simulink

\[ A_{fp} = \text{fi}(A, 1, 32, 10) \]
CONCLUSIONS
Quickly Iterate between Idea and Prototype

- First prototype is functionally correct with automatic C code generation
- Spend your time in optimizing rather than debugging the code
- Find errors reusing the same testbench at each design step
Use MATLAB and Simulink with a variety of hardware platforms for project-based learning. Options range from student-owned hardware to versatile solutions for controls, mechatronics, robotics, and signal processing in classroom labs.

**Arduino**
Student-priced microcontroller board for introducing electrical engineering, motor control, and mechatronics

**Atlys FPGA Development Kit**
Low-cost platform for real-world audio and video applications based on the Xilinx Spartan-6 FPGA

**BeagleBoard**
Low-cost, single-board computer designed for audio, video, and digital signal processing

**Machine Science Microcontroller Kits**
Low-cost kit with Atmel ATmega168 microcontroller for building circuits on a solderless breadboard

**Microchip dsPIC Microcontrollers**
Low-cost boards suitable for controls, signal processing, and custom prototyping

**Quanser**
Hardware experiments, data acquisition cards, and Simulink models for teaching controls

**Classroom Resources**
Find resources for classroom instruction or individual training.

**More Platforms**
- Supported data acquisition devices
- Supported webcams and image acquisition devices
- Supported instruments and instrument interfaces
- Supported Simulink targets
- Example custom targets
- File Exchange at MATLAB Central