LABVIEW FPGA

how to program FPGAs
without any VHDL knowledge
New Labview project: target FPGA

math. vi is no longer targeted to a PC, but to a FPGA
Basic FPGA VI

- $F = (A + B)CD$
LabVIEW Mapped to FPGA

Implementing Logic on FPGA: $F = (A+B)CD$
FPGA palette

FPGA specific functions
- Programming structures
- Device I/O
- Arithmetic and Boolean Logic
- Arrays and clusters
- Timing
- Math and control functions
- Synchronization and FIFOs
- Look-up tables
FPGA palette limitations

Some LabVIEW VIs and functions are not available or have restrictions in FPGA VIs.

The following LabVIEW features are not available for FPGA VIs:

- Floating-point functions
- Variable-size and multidimensional arrays
- Error clusters or strings
- Dialog boxes
- File I/O
- Printing
Integer Math

- No floating point
- Singles and doubles type are not supported
**Integer Math**

- For variable scaling, you can determine and set the scaling factor and bit shift from the host application.

For example: Scaling Factor: 11
Downshift: -4 bits
Result Multiplication: \(11 / 16 = 0.6875\)
Compile Process and Server

- Convert LV diagram to intermediate files
- Send intermediate files to the compile server
  - Compiles VIs for FPGA
  - Returns FPGA bitstream to LabVIEW
  - Bitstream is stored in VI

- LabVIEW environment is a client
  - Can disconnect from server and reconnect while compiling
Download

- Occurs automatically after a compile initiated by the run button

Windows OS
LabVIEW FPGA Module
FPGA VI
Bit File Embedded

Download
Target FPGA
FPGA VI (actually the bit file)
Interactive Mode

- Interact with VI on FPGA through Front Panel
- No Debugging
  - VI is running in the FPGA
Interactive Mode

The VI counts the number of rising edges since the last sample event and measures the total period from the first to last edge in the measurement window. Divide the count by the period to determine the frequency. Press the polarity button to select the number of falling edges. To reset the value of the counter, push the sample button.
Host PC Interactive Mode

- Interact with FPGA through host PC based Front Panel
- Allows you to do other processing in Host VI
Host PC Interactive Mode

Host VI

Host Computer

FPGA VI

FPGA Device
Windows Target Mode

- Run FPGA VI on Windows
- Software Emulation
  - No hardware timing
- Debugging possible
  - Check logic before compile
Benchmark timing performances

- Place code that you want to time here.
- Elapsed Time
  - U32
Exercise: timing benchmark

- Create a VI which adds two numbers and runs a benchmark in parallel that determines how fast code is running
Benchmarkerding VI's Size

- Speed
- Size
  - IOBs – Input/Output Blocks
  - MULT18X18s - multipliers
  - SLICEs – Combination of LookUp Tables (LUTs) and Flip Flops (FFs)
  - BUFGMUXs – portal to the clock net, which is used to clock FFs
Configuring FPGA I/O
Using FPGA I/O Nodes

Two ways to use FPGA I/O:

- Drag and drop from LabVIEW Project
- Drop empty I/O node on block diagram and select I/O by left clicking
Exercise
Timing Control Functions
Configure Timing Functions

- **Counter Units**
  - Ticks
  - \( \mu \text{sec} \)
  - msec

- **Size of Internal Counter**
  - 32 Bit
  - 16 Bit
  - 8 Bit
Timing Using the Single-Cycle Timed Loop

- Execute multiple functions in a single clock cycle
- Loop executes at compile clock speed by default
- Increases code speed and efficiency
- All code must execute within one clock tick

50 MHz Clock = Spartan 3E HW
Multiple Clock Domains

- Derive different clock domains based off 50 MHz
- Different Single-Cycle Timed Loops can have different clocks
- Used for:
  - Generating clocks
  - Local speed optimization
Triggering when multiple digital signals match a logic condition
Designing counters

using the single-cycled timed loop
Designing counters

using the while loop with a wait inside
Parallel Loop Execution

• Dictate Loop Execution Order
  – Structures such as FIFOs and occurrences can determine program flow and loop execution order
  – These structures can also synchronize the execution of parallel loops

• Data Sharing
  – Can pass data between parallel loops on the FPGA
  – Use FIFOs, memory, or local variables
Loop Ordering With FIFOs

- FPGA FIFO passes data between parallel loops
- FIFO determines loop execution order
  - Acquisition writes data to FIFO
  - Read FIFO to display data on indicators
Parallel Execution

• Graphical programming promotes parallel code architectures
• LabVIEW FPGA implements truly parallel execution
1. Xilinx tools: ISE + Chipscope
   Labview compatible

2. Xilinx tools: ISE + Chipscope
   **NO** Labview
ADC interface

RS232 interface

DAC interface

PS2 interface

LCD display

ADC

DAC

RS232

PS2
Task assignment proposal

1. RS232 interface:
   - VHDL
   - Labview (both FW & SW)

2. ADC interface
   - VHDL
   - Labview

3. DAC interface
   - VHDL
   - Labview

4. PS2 interface
   - both ways

5. LCD display
   - your choice