

System-on-Chip FPGA Design Lab

525.742

Doug Wenstrand
Joseph Haber

Class Goals

- To design and develop highly functional systems which reside on a single FPGA
- Effectively merge IP blocks and custom hardware to construct optimal systems
- Efficiently break down computational tasks into hardware and software components for an optimally flexible and high performance system

Grading Criteria

Laboratory Assignments / Homework (75%)

Throughout the semester (not necessarily every week) laboratory assignments will be given. Typically this will give design requirements, and the student will construct a design using a combination of IP cores, custom VHDL, and software. This design will be demonstrated on the student's hardware dev kit (distributed in class). Unless otherwise specified, each assignment will count equally towards the grade. Grading criteria will be specified for each laboratory, typically with intermediate benchmarks that can be demonstrated if the whole assignment is not complete. Some laboratories will have a written component due, some will be only demonstration. In all cases, reliable operation is required, and a clear understanding of what you have done. The instructors will often do a short interview during the demonstration to ask about some of the key design features. **All laboratory assignments from part I of the class will be due before 4/5/2010.**

Final Project (25%)

The final project will be assigned in early April. This will be group projects, with student groups of 2-3 students handling design problems of significant scope. It will be required that the project be broken down into clearly identifiable subsections to allow the tracking of progress of individuals within the group (read: all members of the project will not necessarily get the same grade). **The final project will be presented on the last day of class; with intermediate milestones presented earlier.**

Misc.

Laboratory Assignments are to be completed independently :

Helping each other with tools related problems, class material, or general VHDL knowledge is, of course, allowed and encouraged – copying of projects from other students or from the Web is neither. A notable exception is the first project, for which some of the code comes directly from the tutorial provided to you.

Projects / Homeworks are to be well documented and appropriately designed :

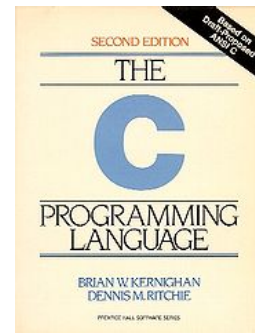
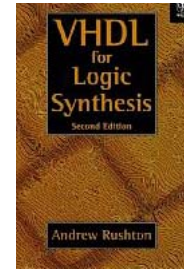
You will note from the published grading breakdown that simply getting the design to meet requirements for demonstration is not grounds for 100%. The design should be done in a well organized, understandable way, with appropriate comments. **After the first assignment, we will often ask that the design should always be accompanied by diagram of your design – this should be developed BEFORE you start to implement your design!** In addition, the design should make effective use of the resources on the chip by implementing pieces in an appropriate manner. The combination of all these factors determines the project grade.

Lecture: As this is a laboratory class, the lecture exists for the main purpose of introducing the concepts in the lab assignment for that week. It is also a suitable forum for general conceptual questions and answers related to labs underway. The rest of the meeting time will be in the laboratory, where both instructors will generally be present to answer more specific questions as needed. Demonstration of completed assignments is also done during this time period.

Homework

- Lab Assignments will *typically* have some or all of these things due
 - Design description (similar to the package one might deliver before a design review) Actual sentences and even paragraphs explaining the design
 - Archived Project Directory (lab1_lastname.zip)
 - Code : readable and commented
 - A “.bit” file, which when loaded to the demonstrator board will produce the final design. Project should build to make above bit file.
 - A Demonstration of the hardware running through a checklist of requirements, along with a brief explanation of the architecture to the instructor.

Class Materials



Book : NO BOOK REQUIRED

VHDL for Logic Synthesis (Andrew Rushton) : good reference for VHDL

The C Programming Language (K&R) : good reference for C language programming

Large Quantity of reference materials will be distributed in class and/or linked from the website.

Additional Materials:

A development kit with board and cables will be provided by the instructor that will contain the FPGA development board and documentation. DVDs with software will also be provided. More on this later.

Course Website :

<http://soc.echelonembedded.com>

Contains class notes and materials that develop as the class progresses.

Class Email Forum :

vhdlforum@echelonembedded.com

by invitation only – if you want an email address “invited” that you didn’t write on the class info form, send it to the instructors @ work and we will invite you.

Class Communication

- Lab corrections and clarifications will be sent to the email forum
 - Make sure to give us an email address that you check *daily*.

Class Expectations

Students

- Will take initiative and read material associated with lab assignments
- Will seek out appropriate reference material from manufacturer data sheets and user manuals
- Will seek to *understand* what they are doing and not pile together pieces in an effort to make things work
- Students will check their email daily

Instructors

- Will provide students with background information and context for each lab
- Will provide suitable direction in the lab assignments
- Will be available for debugging help
- Will provide concrete grading criteria for each lab
- Will respond to forum emails daily (generally much faster)

Debugging Assistance

- Debugging laboratory assignments will almost assuredly be a significant percentage of the time spent on an assignment
- Students are encouraged to seek assistance in debugging when stumped. This can be done in the lab, or by web-submitting an archived project to the instructors and asking for help
 - In order to enforce good troubleshooting skills, the instructor will look only if the student can accurately describe the problem in a specific fashion that shows some effort has already been undertaken
- When an incomplete (non-working) laboratory must be submitted, and explanation of the faults, the possible causes, and the steps taken to identify the problem must be specified.
 - In other words, before the deadline, you may have to cut-bait and spend the remaining 20 minutes getting your project into a presentable (though non-working) state.

Schedule

(subject to change, website is authoritative reference)

Week 1: Introduction / Getting Started. Xilinx Microblaze Intro

Week 2: Audio CODEC, Soft-Core processor peripheral interfacing

Week 3: Soft-Core Processor: Laboratory Session

Week 4: Soft-Core Processor: Additional Topics

Week 5: Soft-Core Processor: Laboratory Session

Week 6: Interfacing With Sensors

Week 7: Signal Processing IP Cores

Week 8: Signal Processing IP Cores: Laboratory Session

Week 9: Networking and Communications Interfaces

Week 10: TCP/IP

Week 11: Final Project Laboratory

Week 12: Additional Topics: Embedded OSes, Final Project Laboratory

Week 13: Final Project Laboratory

Week 14: Final Project Laboratory

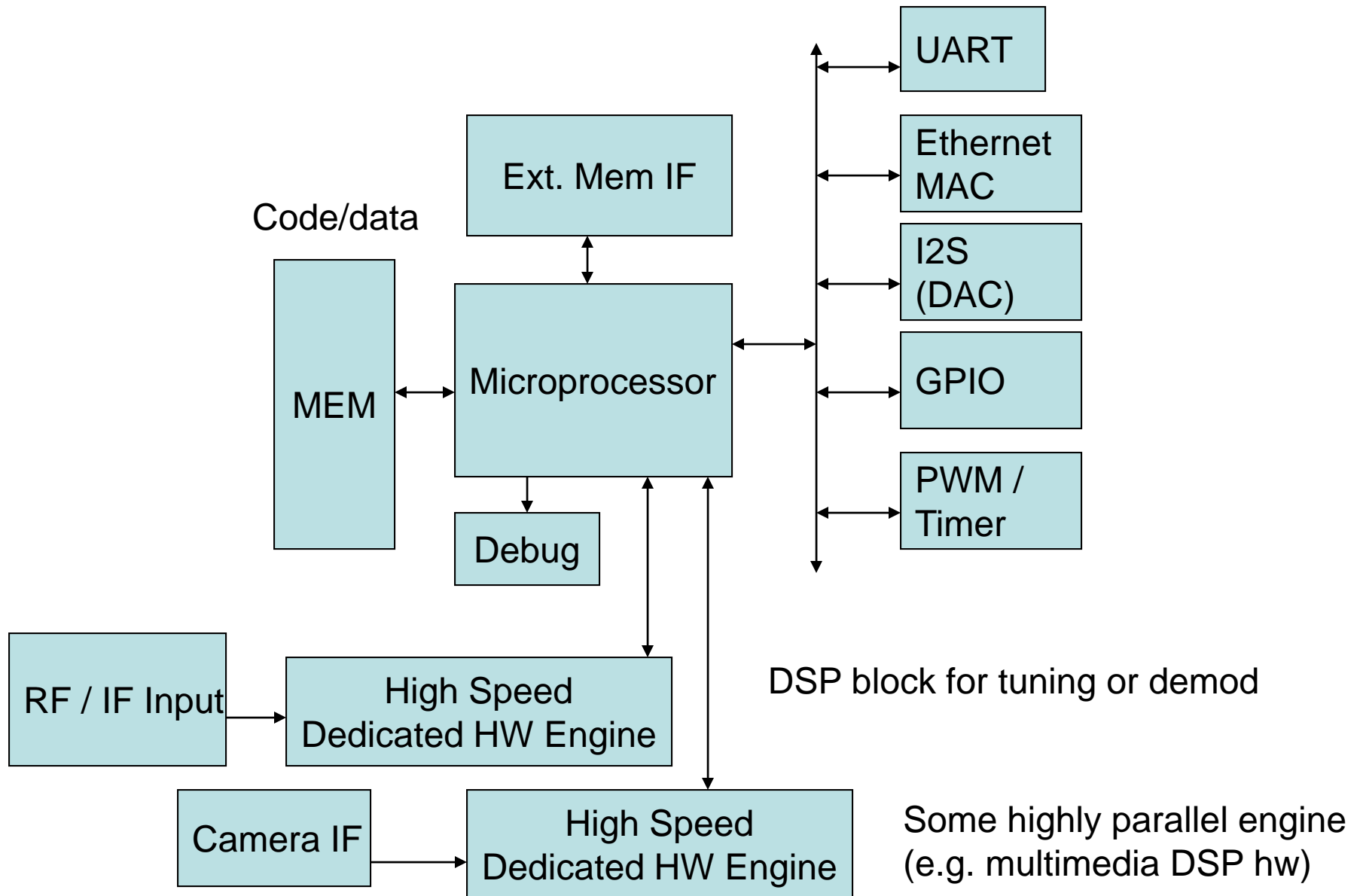
FPGA Gate Counts

- FPGA gate counts have been increasing at Moore's law rates, with early FPGAs in late 80's @ 9k gates, Early 2000s at 1M gates...etc.
- Current FPGAs offer
 - Gate counts in excess of 10 million gates
 - 2000 hardware multipliers
 - 36Mb of memory
 - Many other specialized structures, including clock management circuits and even A/D converters...etc.
- The potential for FPGA penetration into high performance computing tasks is great
 - With these gate counts, no longer does the FPGA have to serve the traditional role of
 - Interface device
 - Dedicated signal processor
 - Glue logic
 - Logic replacement...etc.
 - The FPGA has the features to take over much of the functionality that one might call a “system”

Defining System-on-Chip

- No concrete description or distinction between this term and any highly complex design.
- General principle : integrating multiple components of a system (things that maybe normally would be thought of as separate pieces) into a single IC.
- Often made up of a combination of existing components

An Example SOC



Characteristics of SOC design

- Highly dependent on design re-use / IP
 - FPGA design in past was a part of the job, assigned to a single designer. No one designer is likely going to fill up millions of gates by themselves
 - Use of previously designed blocks or manufacturer provided blocks to build a system
 - More learning to use the tools and understanding other people's work necessary
- Combination of hardware and software design knowledge necessary

Sample blocks for the SOC/FPGA

- Processors
- Memories
- Peripherals
 - PWM
 - Timers
 - Ethernet MAC
 - Custom Peripheral Interfaces
- Hardware Processing Elements
 - Filters
 - FFTs
 - DDS
 - Mixers

FPGA Embedded Processors

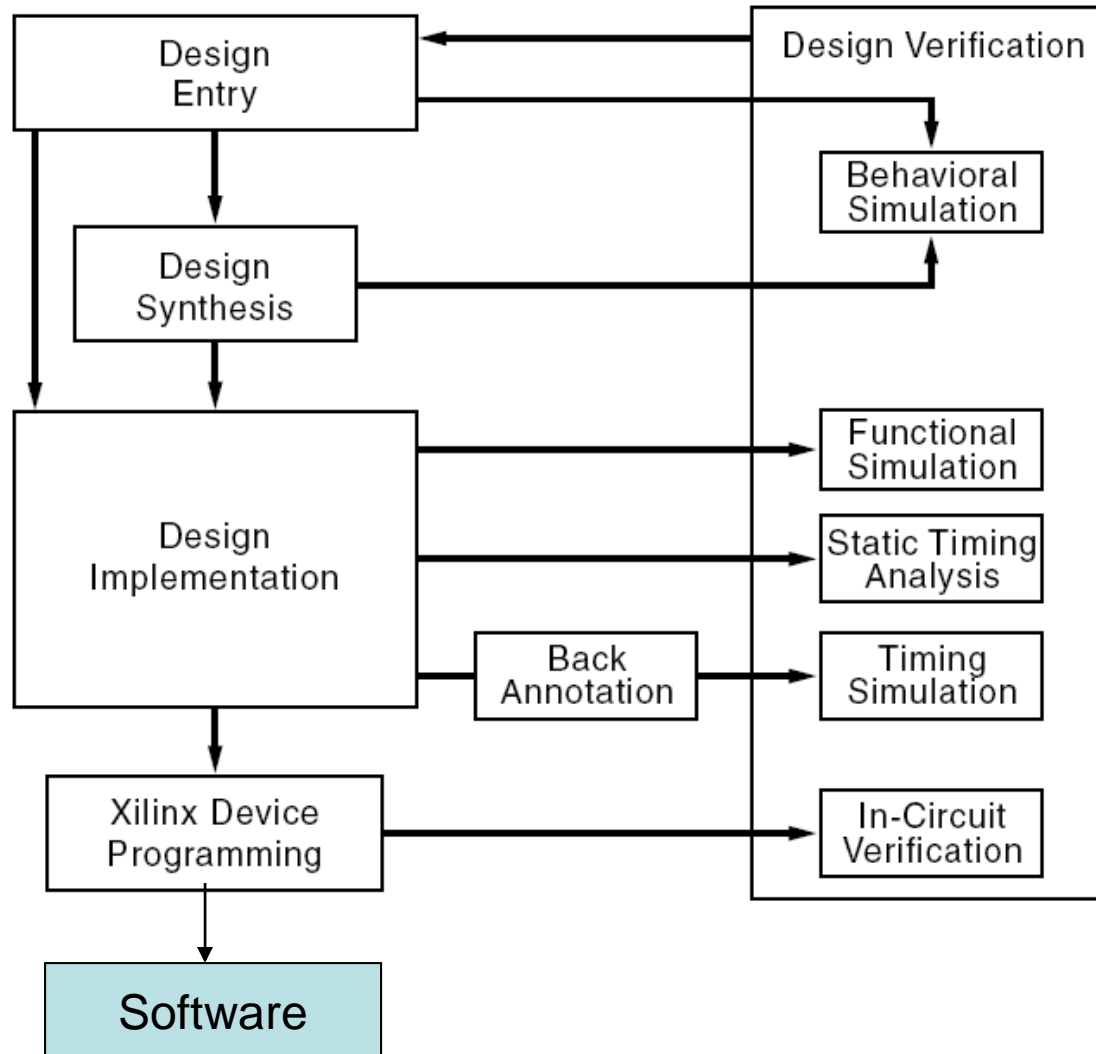
- Soft Core Processor
- A piece of IP (just information) which encapsulates the design of a processor
- Synthesized, placed, routed just like any design you'd make
- Altera NIOS, Xilinx Microblaze, Xilinx Picoblaze, ARM Cortex M1... endless free and for cost choices.
- Hard Core Processor
- FPGA Fabric and traditional microprocessor (i.e. not implemented in the FPGA fabric) together on the same chip
- PowerPC, ARM...etc.
- Generally higher performance
- Not subject to routing...etc. (since this is true silicon, a real data sheet can be made specifying performance)

FPGA Embedded Processors

- In 525.442 emphasis was on **small** embedded processor for simple state-machine tasks
- Larger scale and more capable processors make the benefit of the embedded processor approach far more clear
 - Efficient programming in C
 - Algorithmically complex tasks beyond what can be efficiently coded in hardware design language
 - Software avoids time consuming synthesis step
 - Easier debugging (breakpoints, single step...etc)
 - Legacy code
 - Large bank of available code available

Remember : this is not an argument for making your FPGA into a processor. If your system is just a processor, you've wasted the best part of the FPGA

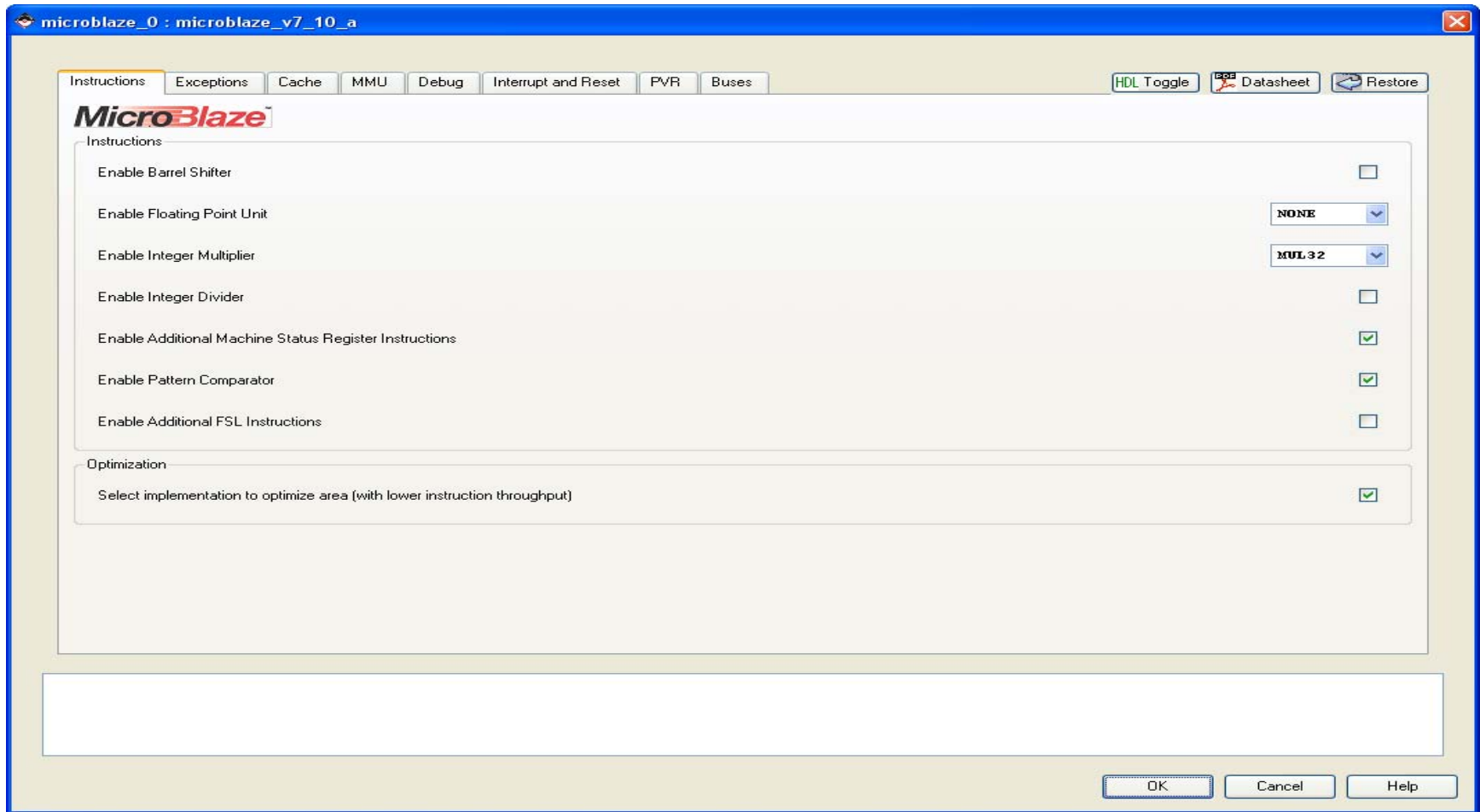
Design Flow with uP Embedded



Xilinx Microblaze

- 32 bit RISC Processor
- Can run up to 106 MHz on the Spartan 3A DSP
- Configurable
 - Cache
 - FPU
 - Data / Program Memory
 - Multiplier
 - High Speed Bus Interfaces
 - Hardware debug interface
- Configuration is done with a graphical tool responsible for setting up and building the Microblaze processor core, together with it's peripherals in an "Embedded Processor" block. This tool is called EDK (Embedded Developers Kit)
 - This general flow is consistent across other FPGA manufacturers.

Xilinx Microblaze



Peripherals in EDK

- DMA engines
- Serial Port, UART, I2C, SPI
- General Purpose I/O
- Ethernet, CAN
- Timers
- Memory Interface

Result : you have built a microprocessor system very much like selecting one out of a catalog. This is the prime benefit of a soft-core system

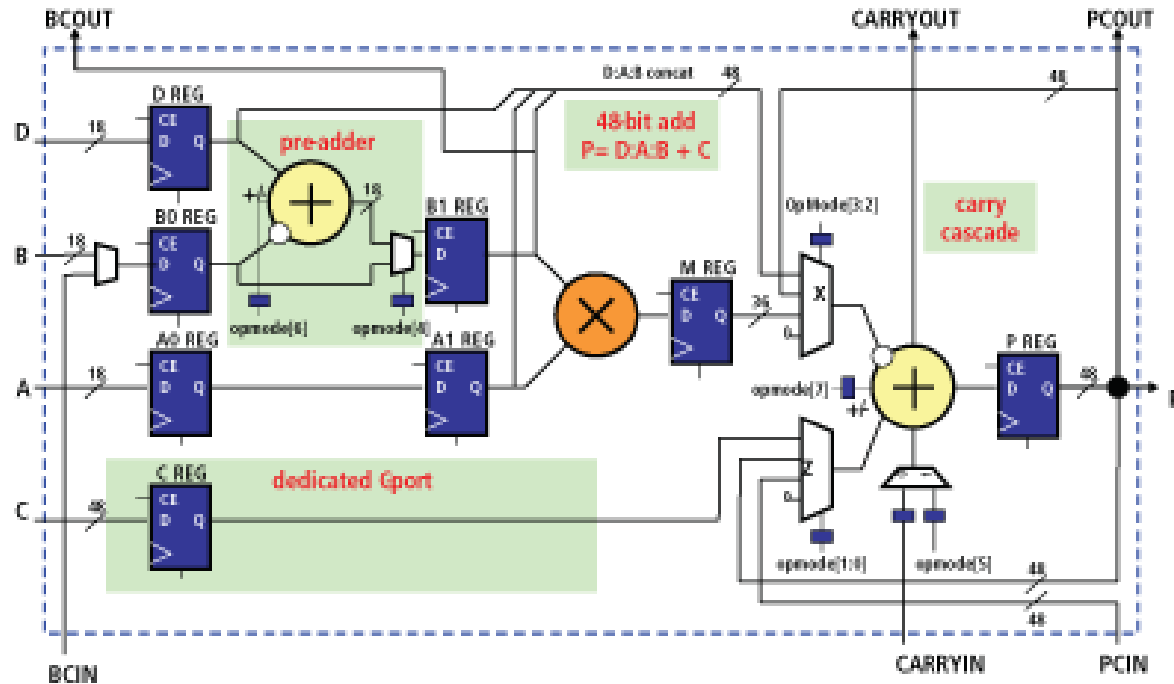
Xilinx Spartan-3A DSP

XC3SD1800A

- 1.8 Million “system gates”
- 16,440 slices (4 LUTs / 4 Flops each)
- Memory
 - 250kbits distributed RAM
 - 1.5Mbits Block RAM
- I/O
 - 309 User I/O pins
 - Wide variety of single-ended and differential standards, with configurable drive strength, integrated termination for differential pairs...etc.
- Clock Management
 - 8 Digital Clock Managers (Frequency Synthesis, Clock De-skewing...etc)
- Target : DSP Applications

Xilinx Spartan-3A DSP

XC3SD1800A



The 250 MHz DSP48A Slice provides an 18-bit x 18-bit multiplier, 18-bit pre-adder, 48-bit post-adder/accumulator, and cascade capabilities for various DSP applications

In addition to the other features, the XC3SD1800 has **84** XtremeDSP48 slices.

Lab Hardware

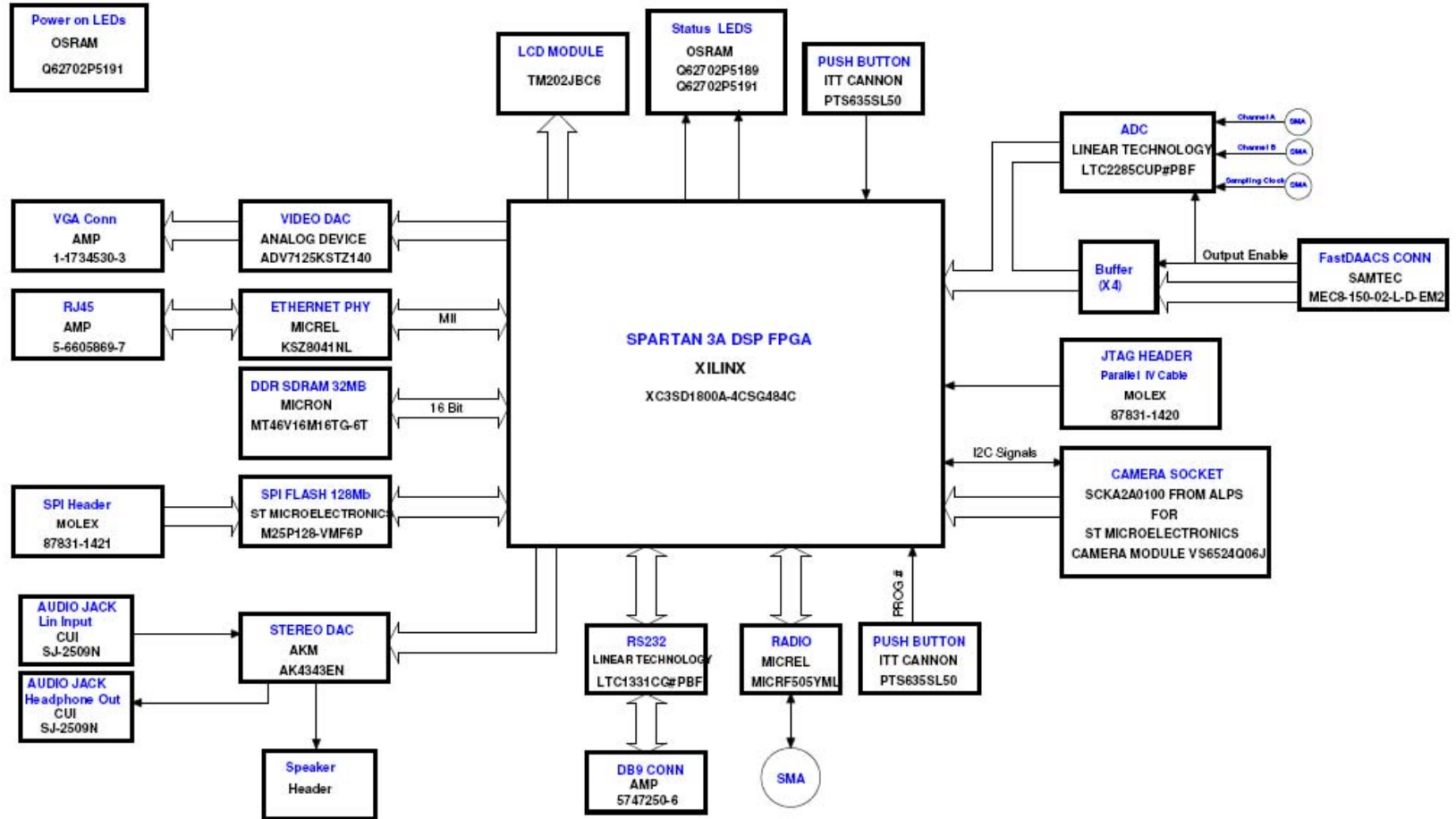
Spartan 3A-DSP Evaluation Kit



Assorted :
Null Modem Cable
Ethernet Crossover Cable
2.5 -> 3.5 mm audio adapter

- FPGA
- Switches
- LEDs
- VGA
- RS232
- Ethernet PHY + Magnetics
- Stereo DAC w/ headphone amp
- LCD Display
- Video DAC
- CMOS color digital camera
- 200kbaud short range wireless transceiver
- 32MB DDR
- 16MB Flash

BLOCK DIAGRAM



Software Toolset

- Xilinx ISE 10.1
 - Synthesis
 - Place / Route
 - Configuration (IMPACT)
 - COREGen
- Modelsim XE
 - Simulation
 - In this class: used primarily on a component by component basis.
 - Used for testing custom pieces, or evaluating cores when documentation isn't sufficient

Software Toolset (cont)

- Xilinx Embedded Development Kit (EDK) 10.1 – Platform Studio
 - Creates a Processor System
 - Menu driven tool that aids in the development of a hardware block containing processor and peripherals
 - Can run as top level
 - entire FPGA design is a processor with peripherals
 - Can run as a subtool to ISE (invoked from within ISE)
 - EDK project is simply another piece of hardware in your top level ISE design
 - ** This is how the class will be oriented
 - Summary : builds a hardware component which is the “processor subsystem” of your FPGA design
- Platform Studio SDK (also part of EDK)
 - Eclipse based Integrated Development Environment
 - Program Editor
 - Compiler
 - Debugger (breakpoints, watch variables...etc)

Week 1 “Lab”

- Pick up Dev Kit and Xilinx Platform Cable
- Install ISE and EDK from provided DVDs
- Step through tutorial on class website
- Complete Tutorial #1 : “Getting Started”
 - A simple LED counting pattern
- Complete Tutorial #2 : “Simple Microblaze Design”
 - Steps through the addition of a processor to your design, and writing some simple software to talk to the serial port, DIP switches.
- Write some software for the Tutorial #2 hardware that implements the requested functionality.
- Lab 1 grading criteria on class website