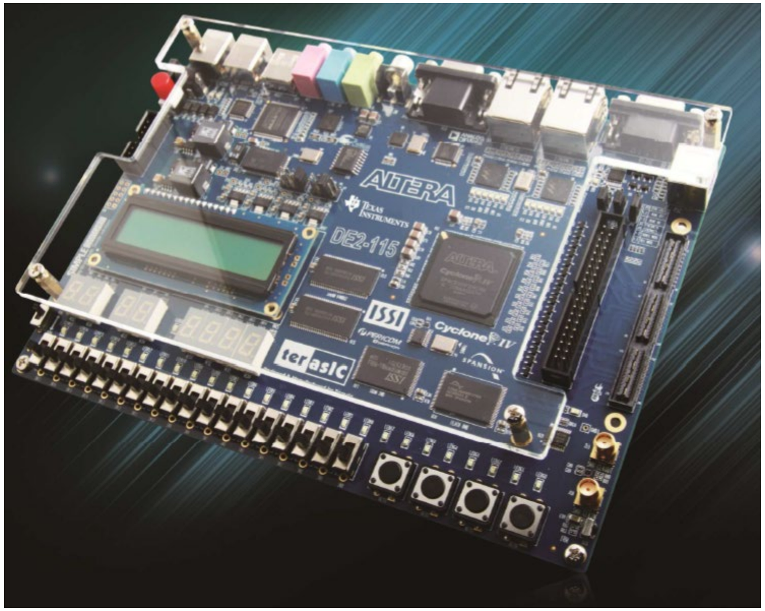
CPE 322 Fundamentals of Hardware Design

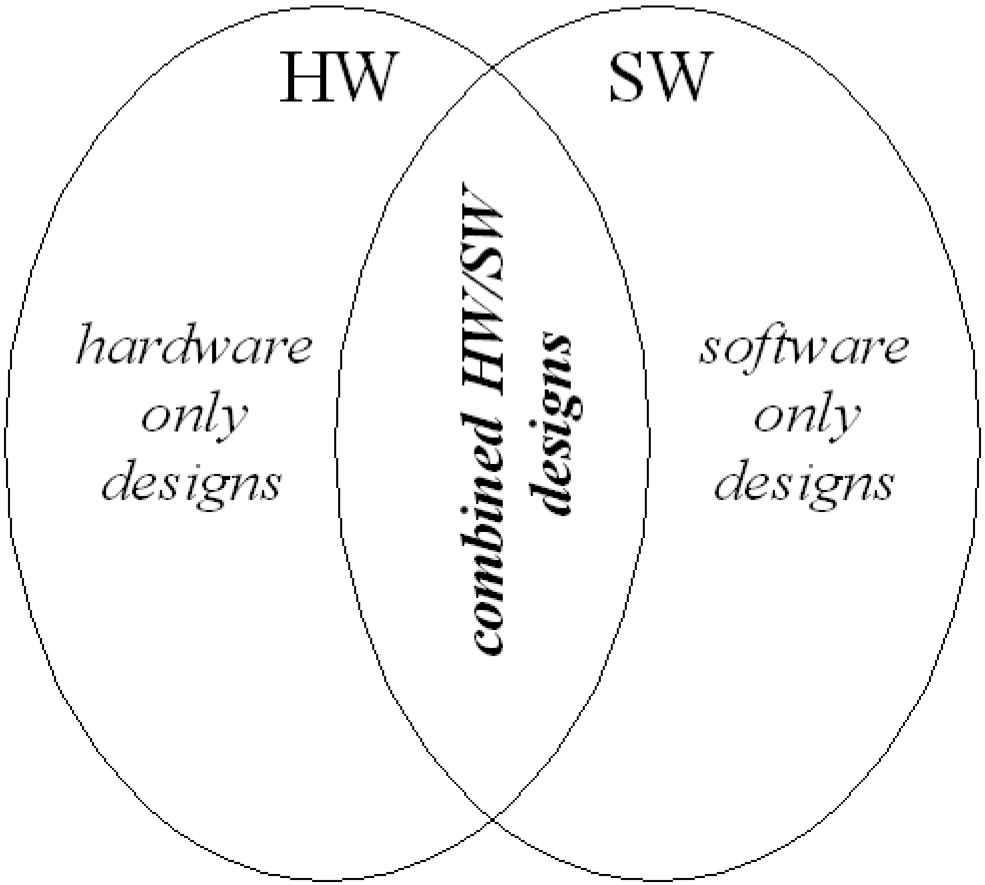
Electrical and Computer Engineering
University of Alabama in Huntsville

Big Picture

General Hardware/Software Design Trade-offs and HW/SW Co-Design Paradigm



Hardware/Software Design Space Continuum



Hardware/Software Trade-offs

- Performance of hardware for a given function can be much faster than a software implementation because
  - There is little or no instruction fetch overhead like there is in a traditional instruction set processor
  - It can make use of the available fine grain concurrency and execute many operations in parallel
- Software is better suited for irregularly-structured control structures
- Traditionally software was considered to be more flexible and field upgradable when compared to hardware
  - FPGA's blurres this distinction though

Hardware/Software Trade-offs

- Cost of implementing a function in hardware is usually much greater than implementing the same function in software
  - Requires additional chip area in an Application Specific Integrated Circuit or board area in a PC board or reconfigurable logic area in an FPGA.
  - Usually takes more time to create a hardware design than a software one
- Both hardware and software resources are limited but hardware resources tend to be more tightly constrained and should be reserved for elements that have the greatest effect on the performance (or possibly other important constraints such as reliability)

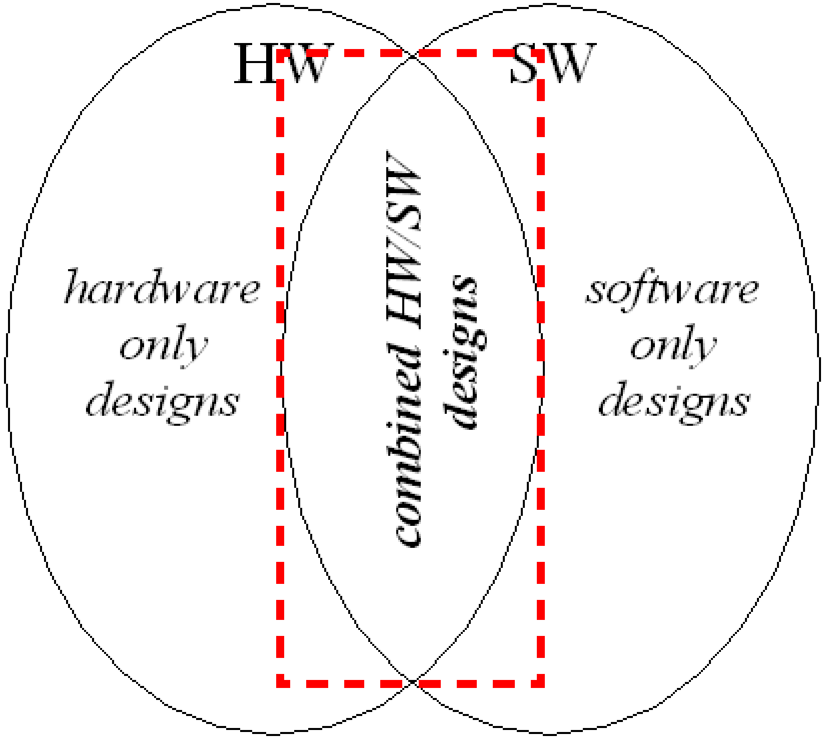
Hardware/Software Trade-offs

- Both hardware and software solutions allow for reduced energy consumption
  - Dynamic and Static Voltage Frequency Scaling, DVFS
  - Dynamic power management
  - Internal Power gating with Data Retention modes
  - Clock Gating (single and multi-level)
- Specific hardware energy consumption strategies can be applied at a finer grain which can result in extremely low power consumption for custom hardware.
- These features are also often dynamically controllable by software at a courser grain using commercially available hardware for much less cost.

Differences between ISP Programming and FPGA Reconfiguration

- Instruction Set Processors, ISPs, such as embedded controllers, allow you to change the program or programs that execute in one or more thread control units.
- FPGAs and other reconfigurable hardware devices allow you to reconfigure both the control units and the data paths of the device.
- In both cases this allows one to increase the usability of the devices involved.
- The allowable number of configurations of internal hardware is much smaller for an ISP when compared to an FPGA but the size of the program (reconfiguration information) is also much smaller!

Embedded Systems



- Application-specific systems which contain hardware and software tailored for a particular task
- Generally part of a larger system
- Responds to external inputs and drives external outputs
- Often must respond in real-time
- Must adhere to strict area (footprint), performance, and power consumption constraints

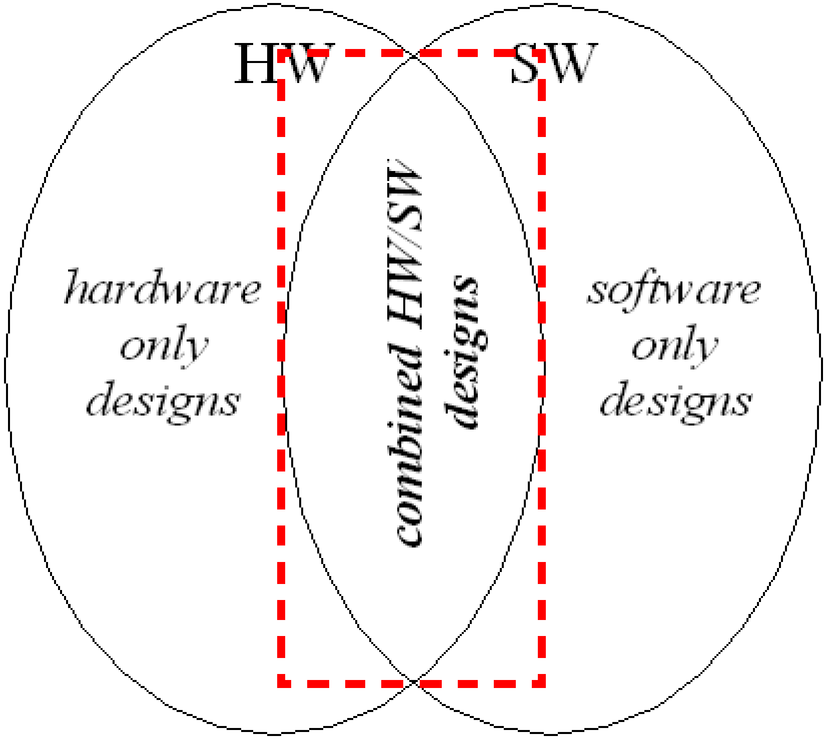
Embedded Systems Examples

- Automotive: Anti-lock breaks, engine control, dynamic ride control, engine diagnostics
- Consumer Electronics: cameras, DVD Controllers, Microwave Ovens, Toasters, Refrigerators, Washers, Smart Card Controllers, RFID systems, Stand Alone GPS systems, TV remote controller, landline telephones, smart scales,
- Industrial Process Controllers: motor control systems , electronic data acquisition and supervisory control, automated laboratory instrument control
- Computer Peripherals: printers, external disk controllers, FAX controllers
- Cell Phones and Peripherals: wireless headsets, wifi bridging devices
- Medical Applications: ECG display and diagnostics, blood cell recorder/ analyzer, patient monitor system
- Network Systems: Routers, network switches, external firewalls

Traditional HW/SW Design Process

- System requirements are developed and then analyzed to determine the “level” of technology required to fulfill these needs.
- Hardware and software teams then independently develop their designs
  - combining of design efforts occur late in the development cycle during the first prototype testing.
- Often leads to a generalized hardware platform being selected and all specialization occurring in the software.

Traditional HW/SW Design Process



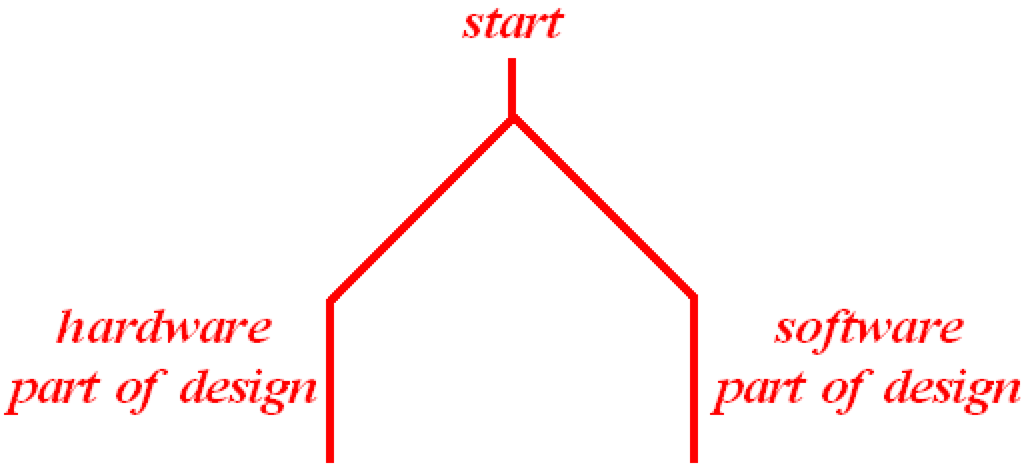
HW

SW

hardware only designs

combined HW/SW designs

software only designs



start

hardware part of design

software part of design

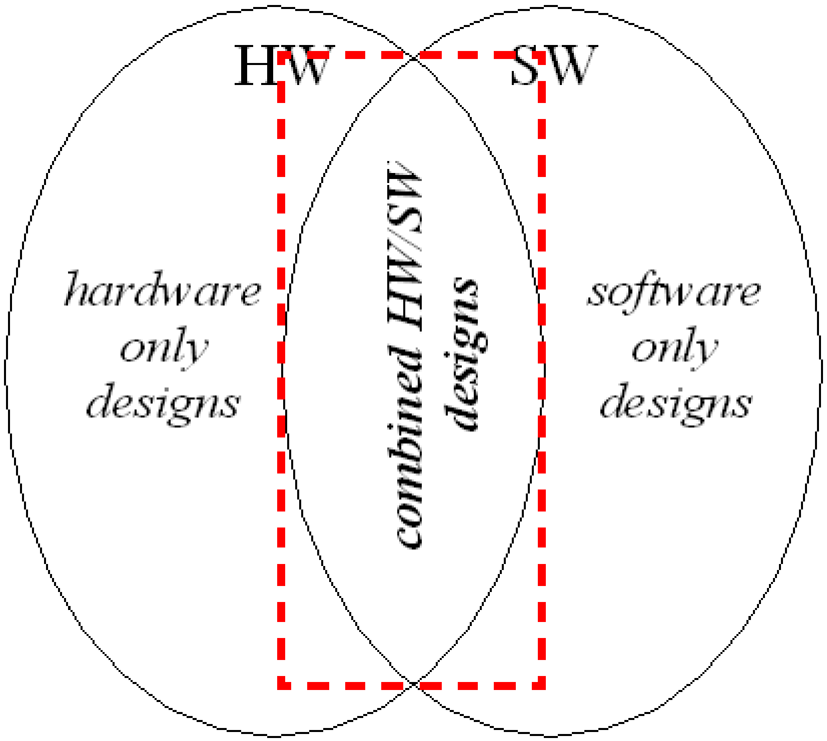
hardware and software portions designed by separate sets of independent experts

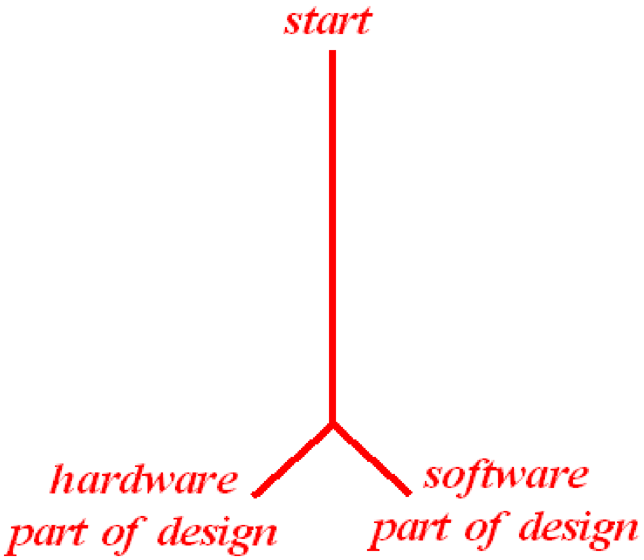
Hardware/Software Co-Design

Meeting system level objectives by exploiting the synergism of hardware and software through their concurrent design.

"Hardware/Software Co-Design", Giovanni De Micheli, and Rajesh K. Gupta, Readings in Hardware/Software Codesign Academic Press, 2002

Hardware/Software Codesign





Hardware and software portions designed by the same group of experts with cooperation with one another. Placement of functionality can occur later in the design cycle

Hardware/Software Co-Design

Meeting system level objectives by exploiting the synergism of hardware and software through their concurrent design.

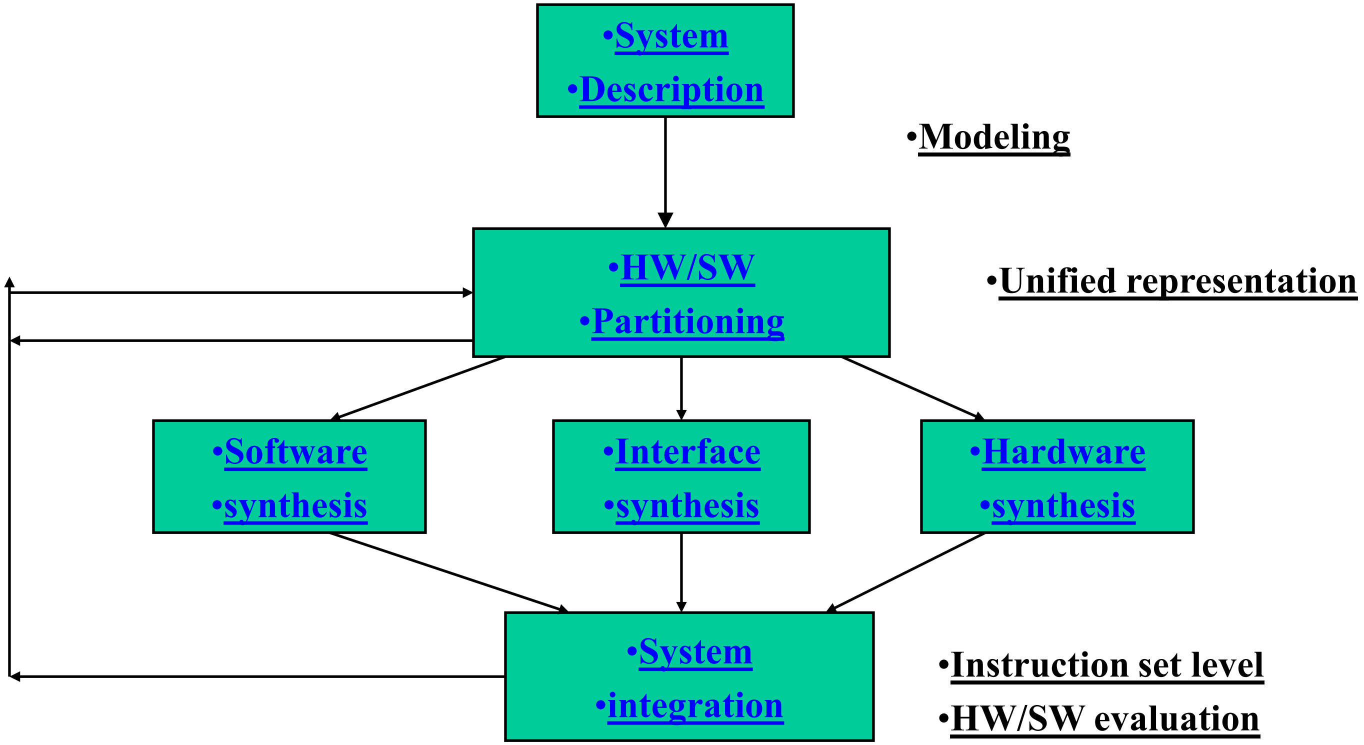
"Hardware/Software Co-Design", Giovanni De Micheli, and Rajesh K. Gupta, Readings in Hardware/Software Codesign Academic Press, 2002

Hardware/Software Co-Design

Design Flow for HW/SW Co-Design

- specification
- modeling
- design space exploration and partitioning
- synthesis and optimization
- validation
- implementation

Hardware/Software Co-Design


•System
•Description
•Modeling
•HW/SW
•Partitioning
•Unified representation
•Software
•synthesis
•Interface
•synthesis
•Hardware
•synthesis
•System
•integration
•Instruction set level
•HW/SW evaluation