EE 3755

Course Summary and Overview

(A) • Computer Arithmetic

— Fixed Point (FXP) Systems: Introduction; Addition; Subtraction.

— Introduction to positional FXP systems for any radix r.

— The special case of r=2 (binary systems).

— Unsigned binary FXP systems; (Dynamic Range or DR).

— Signed binary FXP Systems

- Sign magnitude (SM)
- 1's complement.
- 2's complement.

— Sign magnitude: Definition; DR; zero representation.

— 1's complement: " ; " ; " ; ".

— 2's " : " ; " ; " ; "

- Addition/subtraction in signed binary FXP systems.
- " / " in sign magnitude (SM) system
- " / " in 1's compl. system
- Proof of why when adding in 1's complement system, the carry-out must be ended around and added back to the result.
- Overflows/Underflows for 1's complement system; (what they are and how they are detected)
- Addition/subtraction in 2's compl. system.
- Proof of why when adding in 2's compl. system the carry-out must be ignored.
- Overflows/Underflows for 2's complement system; (what they are and how they get detected; two ways of detection).
- Fixed point (FXP) multiplication.
  - add/shift algorithm for unsigned multiplication
  - Booth algorithms for multiplying signed 2's compl. numbers; (3 algorithms will be presented).

- FXP Division; (Problem definition; definition of division overflow; proof of necessary and sufficient condition for division overflow; shift-subtract/add division algorithm).
- Floating point (FLP) systems; (Introduction; presentation for any radix r; fraction f; exponent e; multiple FLP representations of a #; special case for r=2 (binary) numbers; normalized fractions; calculation of dynamic range (DR); FLP formats (single precision; double presision); biased exponents (what they are and why they are used); overflows/underflows (fraction overflow; fraction underflow; exponent overflow; exponent underflow))
- FLP Addition/Subtraction.
- FLP Multiplication/Division
- Division of fractions (required in FLP division)

— Ripple carry adder (RCA); (simple design but very slow).

— Carry Lookahead Adder (CLA); (very fast adder).

— One-level CLAs; (they require large gates; fan in problem).

— Two-level CLAs; (they solve above problem; a 32-bit two-level CLA will be presented).

— Unsigned array multipliers; (very fast; based on the carry save approach to be explained later). I will cover this topic if time allows. It is very interesting.

— Multiplicative division technique; I will cover this topic if time allows; it is a very interesting topic.

(B) Verilog HDL (Hardware Description Language).

— Verilog basics; (module; (the basic building blocks) module ports; wires; gates; module instatiation; logic value set; binary full adder (explicit descr)

- Expressions; (continuous assignment; operators: bitwise logical; vectors, bit select, part select; sized numbers and constants (parameters); operators: bitwise with vectors; operators: arithmetic; operators: logical AND, OR etc; operators: equality, comparison; operator: conditional; operators: shift, catenate; operator precedence and association; ALU (Arithmetic Logic Uuit Example).
- Delay, Descriptive styles; (delays; Ripple Adder and Delays; the event queue and Verilog Simulation; descriptive styles; (implicit structural desciptive style, behavioral)
- Uusigued and Sigued Adder in Verilog; Construction of ALUs " "
- Carry Lookahead Adders (CLAs); One-Level CLA in Verilog; (5-bit long). Two-" " " " ; (32-" ").

— Procedural Code and behavioral modeling; (procedural code overview; procedural code basics; (initial, always, begin, end, #, $display; always); variables; (registers, integers, etc); event control (@); syntax and simulation of if/else; syntax and simulation of case; syntax and simulation of for, while, repeat, forever loops; ripple adder: combinational vs. sequential; various examples.)

— Integer multiply and divide; (unsigned multiplication in Verilog; Booth algorithms in Verilog; (3 Booth algorithms will be presented); Division in Verilog.

— Floating point; (binary floating-point representation and arithmetic; IEEE 754 FLP standard; FLP Addition H/W in Verilog)

— Synthesis; (synthesis overview; (synthesis are the steps needed to convert a Verilog description into a form that can be manufactured); synthesis steps; synthesis of simple logic; synthesizing arithmetic; synthesis of

conditional operator; synthesis of delays; (7)
synthesis of procedural code (three forms);
synthesis of Form 1 → (combinational Logic
and level-triggered registers (latches));
synthesis of Form 2 → (Edge-triggered flip-flops);
" of conditional class (if/else);
" of " " (case, if/else chains);
" " iteration class (for, while, repeat)).

(C) The MIPS instruction set architecture (ISA);
Design of MIPS Central Processing Unit (CPU) → Design of Data Path
" " Control Unit → Hardwired
→ Micro-programmed.

• MIPS ISA

— MIPS Basics; (Machine language basics;
machine languages covered in ECE courses;
instructions, registers, immediates, memory;
basic three-register instructions;
other basic instructions; instruction coding;
various integer arithmetic instructions;
pseudo instructions.).

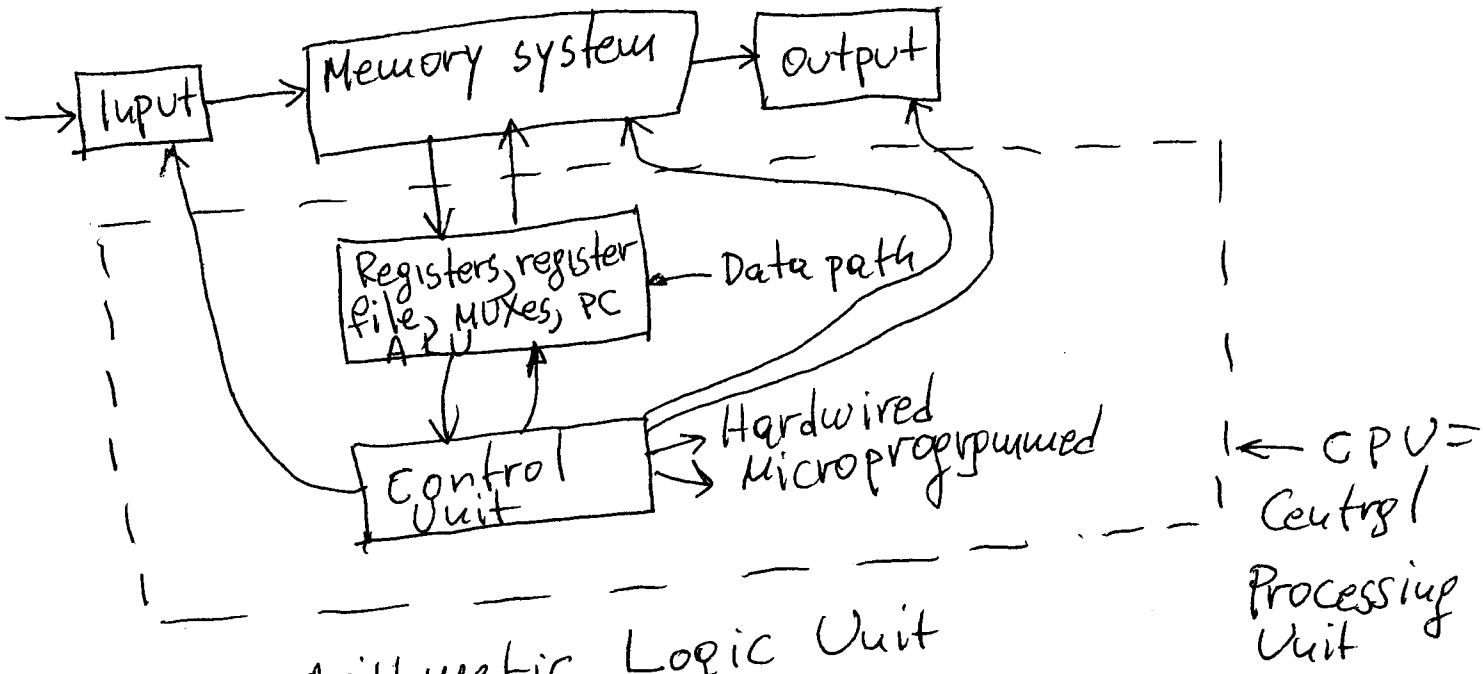
— Control-Transfer Instructions; (types of control-transfer instructions (CCTIs); A simple jump instruction; A branch instruction; more branch instructions; " jump " examples; typical CTI uses)

— Object files and system calls; (structure of SPIM assembly language file; SPIM and real system calls.)

→ Load and Store Instructions; (load byte unsigned; store byte; load byte; load word; store word; load and store half; array access examples; histogram program).

→ More procedure code and compiler; (procedure swap; procedure sort; basics of compiler).

• Design of MIPS CPU → Design of data path
→ " " Control Unit
↙ Hardwired
↘ Microprogrammed.



*. ALU = Arithmetic Logic Unit
MUX = Multiplexor

Figure above is a block diagram of a typical computer.

- MIPS in Verilog ;C (I'll do this if time allows)