Control Unit (single cycle implementation)

- Control unit sends control signals to data path depending
  - on the opcode (and function field)
  - results in the ALU (for example for Zero test)
- These signals control
  - muxes; read/write enable for registers and memory etc.
- Some "control" comes directly from instruction
  - register names
- Some actions are performed at every instruction so no need for control (in this single cycle implementation)
  - incrementing PC by 4; reading instruction memory for fetching next inst.

Building the control unit

- Decompose the problem into
  - Data path control
  - ALU control
- Setting of control lines by control unit totally specified in the ISA
  - for ALU by opcode + function bits if R-R format
  - for register names by instruction
  - for reading/writing memory and writing register by opcode
  - muxes by opcode
  - PC by opcode + result of ALU

Implementation

- Input: opcode
- Output: setting of control lines
- Can be done by logic equations
- If not too many, like in RISC machines
  - Use of PAL's (cf. CSE 370).
  - In RISC machines the control is "hardwired"
- If too large (too many states etc.)
  - Use of microprogramming (a microprogram is a hardwired program that interprets the ISA)
- Or use a combination of both techniques (Pentium)

Where are control signals needed (cf. Figure 5.17)

- Register file
  - Register write signal (R-type, Load)
  - Register destination signal (rd for R-type, rt for Load)
- ALU
  - What kind of second operand (register or immediate)
  - What kind of function (ALU control)
- Data memory
  - Read or write?
  - Control of what is written in result register (from ALU or memory)
- Branch control
  - PC modification if branch is taken

How are the control signals asserted (cf. Fig 5.19)

- Decoding of the opcode by control unit yields
  - Control of the 3 muxes (regdest, ALU 2nd source, source of register write): 3 control lines
  - Signals for register write, read/write of data memory: 3 control lines
  - Signals to activate ALU control (e.g., restrict ourselves to 2)
  - Signal for branch (1 control line)
    - decoding of opcode ANDed with ALU zero result
- Input Opcode: 6 bits
- Output 9 control lines (see Figure 5.27)