Overview of Logic Design

Fundamental Hardware Requirements
- Communication
- How to get values from one place to another
- Computation
- Storage

Bits are Our Friends
- Everything expressed in terms of values 0 and 1
- Communication
  - Low or high voltage on wire
- Computation
  - Compute Boolean functions
- Storage
  - Store bits of information

Digital Signals

- Use voltage thresholds to extract discrete values from continuous signal
- Simplest version: 1-bit signal
  - Either high range (1) or low range (0)
  - With guard range between them
- Not strongly affected by noise or low quality circuit elements
- Can make circuits simple, small, and fast

Computing with Logic Gates

- Outputs are Boolean functions of inputs
- Respond continuously to changes in inputs
  - With some, small delay

- And
  - $a \land b \rightarrow \text{out}$
  - $\text{out} = a \land b$
- Or
  - $a \lor b \rightarrow \text{out}$
  - $\text{out} = a \lor b$
- Not
  - $\neg a \rightarrow \text{out}$
  - $\text{out} = \neg a$

Combinalional Circuits

- Continuously responds to changes on primary inputs
- Primary outputs become (after some delay) Boolean functions of primary inputs

Acyclic Network of Logic Gates

- Primary Inputs
- Primary Outputs

Voltage
- Rising Delay
- Falling Delay
Bit Equality

- Generate 1 if a and b are equal

Hardware Control Language (HCL)
- Very simple hardware description language
- Boolean operations have syntax similar to C logical operations
- We'll use it to describe control logic for processors

Bit equal

![](image)

\[
\text{HCL Expression}
\]

\[
\text{bool } eq = (a\&\&b)||(!a\&\&!b)
\]

Word Equality

Word-Level Representation

- 32-bit word size
- HCL representation
- Equality operation
- Generates Boolean value

Word-Level Representation

- Select input word A or B depending on control signal s
- HCL case expression
- Series of test: value pairs
- Output value for first successful test

Word-Level Representation

- Find minimum of three input words
- HCL case expression
- Final case guarantees match

4-Way Multiplexor

- Select one of 4 inputs based on two control bits
- HCL case expression
- Simplify tests by assuming sequential matching

Arithmetic Logic Unit

- Combinational logic
- Continuously responding to inputs
- Control signal selects function computed
- Corresponding to 4 arithmetic/logical operations in Y86
- Also computes values for condition codes

HCL Word-Level Examples

Minimum of 3 Words

\[
\text{int Min3} = [
\begin{array}{c}
\text{if } A < B \&\& A < C : A; \\
\text{if } A > B \&\& A > C : B; \\
1 : C;
\end{array}
\]

4-Way Multiplexor

\[
\text{int Out4} = [
\begin{array}{c}
\text{if } s1 \&\& s0 : D0; \\
s1 : D1; \\
s0 : D2; \\
1 : D3;
\end{array}
\]

Bit-Level Multiplexor

- Control signal s
- Data signals a and b
- Output a when s=1, b when s=0

HCL Word-Level Examples

- Find minimum of three input words
- HCL case expression
- Final case guarantees match

4-Way Multiplexor

- Select one of 4 inputs based on two control bits
- HCL case expression
- Simplify tests by assuming sequential matching

Arithmetic Logic Unit

- Combinational logic
- Continuously responding to inputs
- Control signal selects function computed
- Corresponding to 4 arithmetic/logical operations in Y86
- Also computes values for condition codes
Storing 1 Bit

Bistable Element

\[ q = 0 \text{ or } 1 \]

Storing 1 Bit (cont.)

Bistable Element

\[ V_n = V_i \]

Storing and Accessing 1 Bit

Bistable Element

\[ q = 0 \text{ or } 1 \]

Physical Analogy

Stable 0

Stable 1

Stable left

Stable right

Metastable

1-Bit Latch

D Latch

Latching

Storing

D

Data

C

Clock

\[ \text{changing D} \]

Transparent 1-Bit Latch

Latching

Changing D

- When in latching mode, combinational propagation from D to Q+ and Q–.
- Value latched depends on value of D as C falls.
**Edge-Triggered Latch**

- Only in latching mode for brief period
- Rising clock edge
- Value latched depends on data as clock rises
- Output remains stable at all other times

**Registers**

- Stores word of data
- Different from program registers seen in assembly code
- Collection of edge-triggered latches
- Loads input on rising edge of clock

**Register Operation**

- Stores data bits
- For most of time acts as barrier between input and output
- As clock rises, loads input

**State Machine Example**

- Accumulator circuit
- Load or accumulate on each cycle

**Random-Access Memory**

- Stores multiple words of memory
- Address input specifies which word to read or write
- Register file
  - Holds values of program registers
  - %eax, %ebp, etc.
  - Register identifier serves as address
  - ID 8 implies no read or write performed
- Multiple Ports
  - Can read and/or write multiple words in one cycle
  - Each has separate address and data input/output

**Register File Timing**

- Reading
  - Like combinational logic
  - Output data generated based on input address
  - After some delay
- Writing
  - Like register
  - Update only as clock rises
Hardware Control Language

- Very simple hardware description language
- Can only express limited aspects of hardware operation
- Parts we want to explore and modify

Data Types

- `bool`: Boolean
  - `a, b, c, ...`
- `int`: words
  - `A, B, C, ...`
  - Does not specify word size—bytes, 32-bit words, ...

Statements

- `bool a = bool-expr ;`
- `int A = int-expr ;`

HCL Operations

- Classify by type of value returned

Boolean Expressions

- Logic Operations
  - `a && b`, `a || b`, `!a`
- Word Comparisons
- Set Membership
  - `A in { B, C, D }`
    - Same as `A == B || A == C || A == D`

Word Expressions

- Case expressions
  - `[ a : A; b : B; c : C ]`
  - Evaluate test expressions `a, b, c, ...` in sequence
  - Return word expression `A, B, C, ...` for first successful test

Summary

Computation

- Performed by combinational logic
- Computes Boolean functions
- Continuously reacts to input changes

Storage

- Registers
  - Hold single words
  - Loaded as clock rises
- Random-access memories
  - Hold multiple words
  - Possible multiple read or write ports
  - Read word when address input changes
  - Write word as clock rises