EE141-Spring 2012

Lecture 8
Wires + Logic

Rikky Muller

Administrativia

- **Midterm on Friday, Feb. 17 9-10:30am in 250 SDH** (TBD)
- Review Session: Thursday Feb. 16, 3:30-5:00pm, 521 Cory (or TBD)
- No Lab on Thursday and next Monday (Presidents day)
- No discussion on Monday

Class Material

- Last lecture
  - Logical Effort + Wires
- Today's lecture
  - Wiring (cont.) – Decoders
- Reading (Ch. 4, 12.3)

Wiring Capacitance

INTERCONNECT

Wire Resistance

$$R = \frac{\rho L}{H W}$$

Sheet Resistance
R_o

$R_1 \equiv R_2$

Interconnect Resistance

Material	ρ (Ω-m)
Silver (Ag)	1.6×10^{-8}
Copper (Cu)	1.7×10^{-8}
Gold (Au)	2.2×10^{-8}
Aluminum (Al)	2.7×10^{-8}
Tungsten (W)	5.5×10^{-8}

Dealing with Resistance

- **Use Better Interconnect Materials**
  - e.g. copper, silicides
- **More Interconnect Layers** 更多的layers可以降低routing的难度。
  - reduce average wire-length
- **Selective Technology Scaling**
  - (More later)

Polycide Gate MOSFET

Silicides: WSi_2, $TiSi_2$, $PtSi_2$ and TaSi

Conductivity: 8-10 times better than Poly

Also applied to source and drain regions

Sheet Resistance

Material	Sheet Resistance (Ω/❑) 面积
n- or p-well diffusion 这个的电容会更大	1000 – 1500
n^+, p^+ diffusion	50 – 150
n^+, p^+ diffusion with silicide	3 – 5
n^+, p^+ polysilicon	150 – 200
n^+, p^+ polysilicon with silicide	4 – 5
Aluminum	0.05 – 0.1

Interconnect Modeling

The Lumped Model

使用条件：the resistive component of the wire is small and the switching frequencies are in the low to medium range

把线对电路造成的延迟全部等分到Cwire中。

$V_{DD} = 1.2 \sim 1\,V$

$I_D = 1mA/mm$

$R_{driver} = \frac{V_{DD}}{I_D}$

忽略了负载电容以及导线电阻，

当导线电阻远小于Rdriver，时，我们可以忽略不计。

$i_{C(t)} = C\frac{dV_{C(t)}}{dt}$

A first approach lumps the total wire resistance of each wire segment into one single R and similarly combines the global capacitance into a single capacitor C. This simple model, called the lumped RC model is pessimistic and inaccurate for long interconnect wires, which are more adequately represented by a distributed rc-model.

The Distributed RC-line

- Analysis method:
  - Break the wire up into segments of length dx
  - Each segment has resistance (r dx) and capacitance (c dx)

The Distributed RC-line

non-branched RC chain (or ladder)

$$I_C = c\Delta L \frac{\partial V}{\partial t} = \frac{(V_{i-1} - V_i) - (V_i - V_{i+1})}{r\Delta L} \longrightarrow \boxed{rc\frac{\partial V}{\partial t} = \frac{\partial^2 V}{\partial x^2}}$$

where V is the voltage at a particular point in the wire, and x is the distance between this point and the signal source.

Each segment has resistance (r dx) and capacitance (c dx)

L represents the total length of the wire, while r and c stand for the resistance and capacitance per unit length

$$\boxed{\tau = \frac{L^2}{2} rc}$$

Wire Model

Model the wire with N equal-length segments:

$$\tau_{DN} = \left(\frac{L}{N}\right)^2 (rc + 2rc + \ldots + Nrc) = (rcL^2)\frac{N(N+1)}{2N^2} = RC\frac{N+1}{2N}$$

The resistance and capacitance of each segment are given by rL/N and cL/N.And using formula， we can compute the dominant time constant of the wire as above.

For large values of N:

$$\tau_{DN} = \frac{RC}{2} = \frac{rcL^2}{2}$$

rc delays should only be considered when the rise (fall) time at the line input is smaller than RC, the rise (fall) time of the line.
t_rise<RC
with R and C the total resistance and capacitance of the wire. When this condition is not met, the change in signal is slower than the propagation delay of the wire, and a lumped capacitive model suffices

RC-Models

Some of the important reference points in the step response of the lumped and the distributed RC model of the wire are tabulated in this 4.7

不同时间点的延迟变量名称不一样。

Voltage Range	Lumped RC-network	Distributed RC-network
0→50% (t_p)	0.69 RC	0.38 RC
0→63% (τ)	RC	0.5 RC
10%→90% (t_r)	2.2 RC	0.9 RC

Step Response of Lumped and Distributed RC Networks: Points of Interest.

这些都是wire的RC model，可以用于wire的延迟分析，其中pai model已经在CAD中学过了。

从上往下，RC model越来越复杂，分析越来越麻烦，但是分析的准确性或精确度越来越高。

Driving an RC-line

Elmore Delay τ: Tree Walk Computing Recipe

- **Do this:**
  - Set τ **= 0**; walk down path of resistors from **Root** to **Leaf** where you want delay
  - At each resistor, do $\tau = \tau + R \cdot \sum$ **(all capacitors downstream)**
  - **"Downstream capacitor"** = any **C** that is reachable in tree below this resistor

Example: at **Ri=2** resistor in our RC tree, the term we would add to **Elmore delay** τ on a tree walk through Ri **= 2×(2+1+1+1+3)**

$$\tau_D = R_s C_w + \frac{R_w C_w}{2} = R_s C_w + 0.5 r_w c_w L^2$$

Rw = rL and Cw = cL
The delay introduced by the wire resistance becomes dominant when (RwCw)/2 ≥ RsCw, or L ≥ 2Rs/r.

因为上面的推理表明电线的电阻只有一半数值是会影响到整个电路的延迟的。

It is assumed here that the driving gate can be modeled as voltage source with a finite source resistance Rs.

$$t_p = 0.69 R_s C_w + 0.38 R_w C_w$$

根据表格查找到对应的两个模型的延迟值并相加。

Resistivity and Performance

Diffused signal propagation

Delay ~ L^2

This figure shows the response of a wire to a step input, plotting the waveforms at different points in the wire as a function of time.

The Global Wire Problem

$$T_d = 0.377 R_w C_w + 0.693\left(R_d C_{out} + R_d C_w + R_w C_{out}\right)$$

Yet, delay is bound to increase substantially for intermediate and long wires, independent of the scaling scenario. To keep these delays from becoming excessive, interconnect technology has to be drastically improved. One option is to use better interconnect (Cu) and insulation materials (polymers and air). The other option is to differentiate between local and global wires. In the former, density and low-capacitance are crucial, while keeping the resistance under control is crucial in the latter. To address these conflicting demands, modern interconnect topologies combine a dense and thin wiring grid at the lower metal layers with fat, widely spaced wires at the higher levels, as is illustrated in Figure 4.28. Even with these advances, it is obvious that interconnect will play a dominant role in both high-performance and low-energy circuits for years to come

Challenges

- No further improvements to be expected after the introduction of Copper (superconducting, optical?)
- Design solutions
  - Use of fat wires
  - Efficient chip floorplanning
  - Insert repeaters

A straightforward approach is to scale all dimensions of the wire by the same factor S as the transistors (ideal scaling). This might not be possible for at least one dimension of the wire, being the length. It can be surmised that the length of local interconnections — wires that connect closely grouped transistors — scales in the same way as these transistors. On the other hand, global interconnections, that provide the connectivity between large modules and the input-output circuitry, display a different scaling behavior.

A constant delay is predicted for local wires, while the delay of the global wires goes up with 50% per year (for S = 1.15 and SC = 0.94). This is in great contrast with the gate delay, which reduces from year to year.

Interconnect: # of Wiring Layers

0.25 μm wiring stack

Figure 4.28

# of metal layers is steadily increasing due to:

- Increasing die size and device count: we need more wires and longer wires to connect everything
- Rising need for a hierarchical wiring network; local wires with high density and global wires with low RC

Using Bypasses

Sometimes, it is hard to avoid the use of long polysilicon wires. A good example of such circumstance are the address lines in memories, which must connect to a large number of transistor gates. Keeping the wires in polysilicon increases the memory density substantially by avoiding the overhead of the extra metal contacts. The polysilicon-only option unfortunately leads to an excessive propagation delay. One possible solution is to drive the word line from both ends, as shown in Figure 9.17a. This effectively reduces the worst-case delay by a factor of four. Another option is to provide an extra metal wire, called a bypass, which runs parallel to the polysilicon one, and connects to it every k cells (Figure 9.17b). The delay is now dominated by the much shorter polysilicon segments between the contacts. Providing contacts only every k cells helps to preserve the implementation density

(a) **Driving a word line from both sides**

(b) **Using a metal bypass**

Figure 9.17 Approaches to reduce the word-line delay

Reducing RC-delay Using Repeaters

The most popular design approach to reducing the propagation delay of long wires is to introduce intermediate buffers, also called repeaters, in the interconnect line

Repeater

$m_{opt} = L\sqrt{\frac{0.38rc}{t_{pbuf}}} = \sqrt{\frac{t_{pwire(unbuffered)}}{t_{pbuf}}}$; $t_{p,opt} = 2\sqrt{t_{pwire(unbuffered)} t_{pbuf}}$

Repeaters

Making an interconnect line m times shorter reduces its propagation delay quadratically, and is sufficient to offset the extra delay of the repeaters when the wire is sufficiently long. Assuming that the repeaters have a fixed delay t_{pbuf} we can derive the delay of the partitioned wire.

$t_p = 0.38rc\left(\frac{L^2}{m}\right)m + m t_{pbuf}$

对m求导得到最小值

The above analysis is simplified and optimistic in the sense that tpbuf is a function of the load capacitance. Sizing the repeaters is needed to reduce the delay. A more precise expression of the delay of the interconnect chain is obtained by modeling the repeater as an RC network, and by using the Ellmore delay approach. Assuming that Rd and Cd are the resistance and capacitance of a minimum-sized repeater, and s is the sizing factor, this leads to the following expression:

$$t_p = m\left(\boxed{0.69\frac{R_d}{s}\left(s\gamma C_d + \frac{cL}{m} + sC_d\right)} + 0.69\left(\frac{rL}{m}\right)(sC_d) + 0.38rc\left(\frac{L}{m}\right)^2\right)$$

$$m_{opt} = L\sqrt{\frac{0.38rc}{0.69R_dC_d(\gamma+1)}} = \sqrt{\frac{t_{pwire(unbuffered)}}{t_{p1}}}$$

$$s_{opt} = \sqrt{\frac{R_dc}{rC_d}}$$

$t_{p,min} = 2.4L\sqrt{R_dC_drc}$

Repeater Insertion (Revisited)

Taking the repeater loading into account

$$m_{opt} = L\sqrt{\frac{0.38rc}{0.69R_dC_d(\gamma+1)}} = \sqrt{\frac{t_{pwire(unbuffered)}}{t_{p1}}}$$

$$s_{opt} = \sqrt{\frac{R_dc}{rC_d}}$$

For a given technology and a given interconnect layer, there exists an optimal length of the wire segments between repeaters. The delay of these wire segments is independent of the routing layer!

$$L_{crit} = \frac{L}{m_{opt}} = \sqrt{\frac{t_{p1}}{0.38rc}} \qquad t_{p,crit} = \frac{t_{p,min}}{m_{opt}} = 2\left(1+\sqrt{\frac{0.69}{0.38(1+\gamma)}}\right)t_{p1}$$

It can be derived that the delay of a segment of critical length always equals 3.23RdCd, and is independent of the routing layer. Inserting repeaters to reduce the delay of a wire only makes sense when the wire is at least two times longer than the critical length.

Even with buffer insertion, the delay of a resistive wire cannot be reduced below the minimum dictated by Equation tp,opt The wire delay all-by-itself hence becomes the limiting factor on the performance achievable by the integrated circuit. The only way to address this bottleneck is to tackle it at the system architecture-level.Wire pipelining is a popular performance-improvement technique in this category.

Example: Memory Decoders

Random Access Chip Architecture

- Conceptual: linear array
  - Each box holds some data
  - But this does not lead to a nice layout shape
  - Too long and skinny

- Create a 2-D array
  - Decode Row and Column address to get data

Memory Architecture: Decoders

Intuitive architecture for N x M memory
Too many select signals:
N words == N select signals

Decoder reduces the number of select signals
$K = log_2 N$

Memory Architecture

CORE:
- keep square within a 2:1 ratio
- rows are **word lines**
- columns are **bit lines**
- data in and out on columns

DECODERS:
- needed to reduce total number of pins; N+M address lines for 2^{N+M} bits of storage

Ex: if N+M=20 → 2^{20}= 1Mb

MULTIPLEXING:
- used to select one or more columns for input or output of data

Row Decoders

Collection of 2^M complex logic gates
Organized in regular and dense fashion

(N)AND Decoder

$$WL_0 = A_0 A_1 A_2 A_3 A_4 A_5 A_6 A_7 A_8 A_9$$

$$WL_{511} = \bar{A}_0 A_1 A_2 A_3 A_4 A_5 A_6 A_7 A_8 A_9$$

NOR Decoder

$$WL_0 = \overline{A_0 + A_1 + A_2 + A_3 + A_4 + A_5 + A_6 + A_7 + A_8 + A_9}$$

$$WL_{511} = \overline{A_0 + \bar{A}_1 + \bar{A}_2 + \bar{A}_3 + \bar{A}_4 + \bar{A}_5 + \bar{A}_6 + \bar{A}_7 + \bar{A}_8 + \bar{A}_9}$$

Decoder Design Example

- Look at decoder for 256x256 memory block (8KBytes)

Problem Setup

- Goal: Build fastest possible decoder with static CMOS logic

- What we know
  - Basically need 256 AND gates, each one of them drives one word line

Problem Setup (1)

- Each word line has 256 cells connected to it
- Total output load is $256*C_{cell} + C_{wire}$
- Assume that decoder input capacitance is $C_{address}=4*C_{cell}$
- Each address drives $2^8/2$ AND gates
  - A0 drives ½ of the gates, A0_b the other ½ of the gates
- Neglecting C_{wire}, the fan-out on each one of the 16 address wires is:

$$BF = \frac{C_{load}}{C_{in}} = \frac{(2^8/2)\left(2^8 C_{cell}\right)}{4C_{cell}} = 2^{13}$$

Decoder Fan-Out

- FB of at least 2^{13} means that we will want to use more than $\log_4(2^{13}) = 6.5$ stages to implement the AND8

- Need many stages anyways
  - So what is the best way to implement the AND gate?
  - Will see next that it's the one with the most stages and least complicated gates

8-Input AND

LE=10/3 1	*LE*=2 5/3	*LE*=4/3 5/3 4/3 1
ΠLE = 10/3	ΠLE = 10/3	ΠLE = 80/27
P = 8 + 1	P = 4 + 2	P = 2 + 2 + 2 + 1

8-Input AND

- Using 2-input NAND gates
  - 8-input gate takes 6 stages
- Total LE is $(4/3)^3 \approx 2.4$
- So PE is $2.4*2^{13}$ – optimal N of ~7.1

Decoder So Far

- 256 8-input AND gates
  - Each built out of tree of NAND gates and inverters
- Issue:
  - Every address line has to drive 128 gates (and wire) right away
  - Can't build gates small enough - Forces us to add buffers just to drive address inputs

Look Inside Each AND8 Gate

Predecoders

- Use a single gate for each of the shared terms
  - E.g., from A_0, $\overline{A_0}$, A_1, and $\overline{A_1}$, generate four signals: A_0A_1, $\overline{A_0}A_1$, $A_0\overline{A_1}$, $\overline{A_0}\overline{A_1}$
- In other words, we are decoding smaller groups of address bits first
  - And using the "predecoded" outputs to do the rest of the decoding

Predecoder and Decoder

Hierarchical Decoders

Multi-stage implementation improves performance

NAND decoder using 2-input pre-decoders

The Next Question

- Impact of wiring!