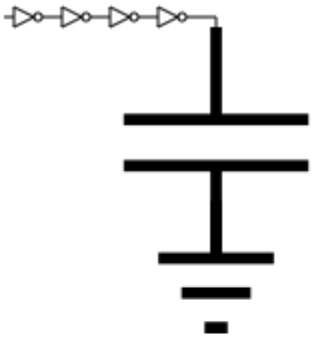
3. Method of Logical Effort

In this chapter, we introduce the *method of logical effort* to design fast digital circuits. This method is based on the ***Model of Logical Effort*** for the delay of digital circuits. The method of logical effort is sufficiently accurate and simple enough to facilitate paper-and-pencil design of fast circuits. **Experiment 3.1** offers a first glance at the problem of designing a fast inverter chain.

▶ **Experiment 3.1:** Delay of 4-Inverter Path

The digital circuit below is a chain of four inverters and a load capacitance, which represents the input capacitance of one or more logic gates. The delay of the circuit is the time it takes the inverter chain to charge or discharge the load capacitance after the input changes. This delay is proportional to the transistor gate capacitances and, therefore, the size of the inverters and the load. You can vary the scale factors of the inverters and the size of the load with the sliders below the schematic. The stage-1 inverter is a reference inverter. Try varying the scale factors of the stage-2, stage-3, and stage-4 inverters to minimize the delay for a given load.



Inverter 2: 1 Inverter 3: 1 Inverter 4: 1

Load: 300 Delay: 107

For a capacitive load of 1200 units, you should be able to reduce the delay from 407 time units when using reference inverters with scale factor 1 by a significant factor (close to 20) when choosing the inverter sizes properly. In fact, gate scaling enables us to reduce the delay to its square root.

If **Experiment 3.1** makes you wonder whether there is a guiding principle to design a fast circuit, this chapter should satisfy your curiosity. We study the delay of digital circuits, beginning with a single gate. Then, we consider circuit topologies with gradually increasing complexity, and introduce techniques for analyzing their delay. As a whole, these techniques constitute the **method of logical effort**, perhaps best perceived as a way of thinking about the design of fast digital circuits. The hallmark of the method is not merely its use for timing analysis but the insight it provides on how to minimize the delay of a circuit. Ivan Sutherland and Bob Sproull, two contemporary computer scientists, pioneered the method of logical effort **[SSH99]**. In this chapter, we introduce the method of logical effort in a refined form.

Ivan Sutherland

Bob Sproull

3.1. Delay of Logic Gates

The ***Model of Logical Effort*** defines the dimensionless delay of a logic gate as

$$d = gh + p,$$

where g is the logical effort, h is the electrical effort, and p is the parasitic delay.

Logical effort and parasitic delay are gate specific, and are constant for a particular gate. For example, a reference inverter has logical effort $g_{inv} = 1$ and parasitic delay $p_{inv} = 1$. These values do not change if we scale the reference inverter in size by increasing its transistor widths. An inverter that is twice as large as the reference inverter still has logical effort $g_{inv} = 1$ and parasitic delay $p_{inv} = 1$.

In contrast, the electrical effort of a logic gate depends on its size. If a gate with input capacitance C_{in} drives capacitive load C_L, then the electrical effort of the gate is

$$h = \frac{C_L}{C_{in}}.$$

Because input capacitance C_{in} is proportional to the size of the logic gate, the larger the gate, the smaller electrical effort h. Consider the reference inverter with normalized input capacitance $C_{inv} = 3$. If we scale the reference inverter with scale factor $\gamma \geq 1$, then the scaled inverter has input capacitance $C_{in} = \gamma C_{inv} = 3\gamma$. The electrical effort of the inverter is

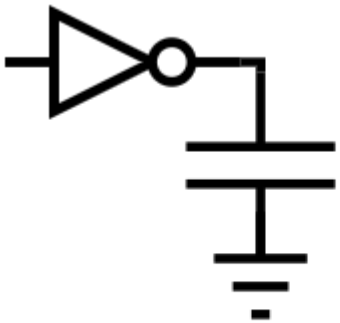
$$h = \frac{1}{\gamma}\frac{C_L}{C_{inv}},$$

which is by factor γ smaller than the electrical effort of the reference inverter with the same load. In a digital circuit consisting of an inverter driving capacitive load C_L, the delay of the inverter is

$$d = h + 1 = \frac{C_L}{3\gamma} + 1. \tag{1}$$

Experiment 3.2 enables you to choose load C_L, and vary the size of the inverter by means of scale factor γ in range $1 \le \gamma \le 10$. Observe that for a given load, increasing the size of the inverter reduces the delay according to Equation **(1)**.

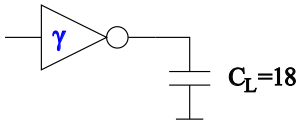
▶ **Experiment 3.2:** Delay of Inverter with Load



Inverter: 6 Load: 15 Delay: 1.83

The delay of the inverter in **Experiment 3.2** is a linear function of the electrical effort. When designing circuits, we may vary the electrical effort of a logic gate by gate sizing or by changing its load. **Figure 1.45** shows the delay of an inverter as a function of electrical effort h. Also shown in **Figure 1.45** is the delay of a NAND gate. Since the NAND gate has a logical effort of $g_{nand} = 4/3$, its delay grows faster with increasing h than for the inverter. The delay of a NOR gate with $g_{nor} = 5/3$ increases even faster with increasing h than the delay of the NAND gate.

Compare the delay of the reference inverter with load $C_L = 18$ capacitive units to that of a scaled inverter with $\gamma = 4$.



a. Compute the delay of the reference inverter ($\gamma = 1$).
b. Compute the delay of the scaled inverter for $\gamma = 4$.
c. Use **Experiment 3.2** to verify the delays in (a) and (b).

The delay of a reference inverter is $d = g_{inv}h + p_{inv} = h + 1$, because $g_{inv} = 1$ and $p_{inv} = 1$ by definition. Electrical effort $h = C_L/C_{inv} = 18/3 = 6$, because the reference inverter has input capacitance $C_{in}(inv) = C_{inv} = 3$. We conclude that the delay of the reference inverter driving load $C_L = 18$ is

$$d_{\gamma=1} = \frac{18}{3} + 1 = 7$$

time units.

If we scale the inverter with scale factor $\gamma = 4$, the pMOS transistor has normalized width $W_p = \gamma \cdot 2 = 8$ and the nMOS transistor $W_n = \gamma \cdot 1 = 4$ units. Thus, the input capacitance is scaled by γ to become $C_{in} = \gamma \cdot C_{inv} = 4 \cdot 3 = 12$. As a consequence, the electrical effort to drive load $C_L = 18$ decreases to $h = 18/12 = 3/2$, and the delay of the scaled inverter is

$$d_{\gamma=4} = \frac{18}{12} + 1 = \frac{5}{2}$$

time units. Increasing the size of the inverter by a factor of four decreases its delay by a factor of $14/5$, or almost by a factor of three.

We can verify both delays in **Experiment 3.2** by adjusting the load to 18 capacitive units, and then observing the delays by adjusting the inverter scale factor to $\gamma = 1$ and $\gamma = 4$.

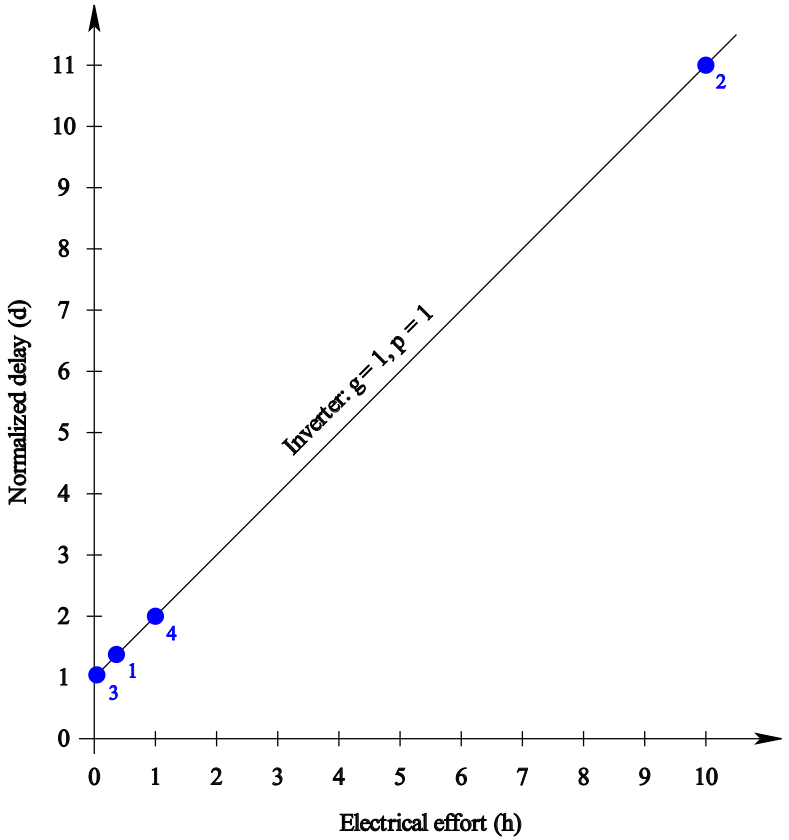
Hide

Draw the delay points at the boundaries of the slider ranges of inverter scale factor and load in **Experiment 3.2** in a $d(h)$-diagram as shown in **Figure 1.45**.

Use **Experiment 3.2** to determine the delays for scale factors $\min \gamma = 1$ and $\max \gamma = 10$ and loads $\min C_L = 1$ and $\max C_L = 30$. The corresponding pairs of electrical efforts $h = C_L/\gamma C_{inv} = C_L/3\gamma$ and delays are:

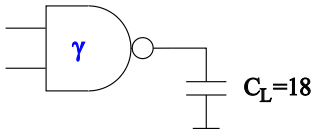
	γ	C_L	h	d
1	1	1	0.33	1.33
2	1	30	10	11
3	10	1	0.03	1.03
4	10	30	1	2

The $d(h)$-diagram below shows the four delay points on the inverter line.



Hide

Compare the delay of the matched NAND gate with load $C_L = 18$ capacitive units to that of a scaled NAND gate with $\gamma = 6$.



a. Compute the delay of the matched NAND gate ($\gamma = 1$).
b. Compute the delay of the scaled NAND gate for $\gamma = 6$.
c. Draw the delay points of (a) and (b) in a $d(h)$-diagram as shown in **Figure 1.45**.

The delay of a matched 2-input NAND gate is $d = g_{nand2}h + p_{nand2}$, where $g_{nand2} = 4/3$ and $p_{nand2} = 2$. The electrical effort of the NAND gate with load $C_L = 18$ is $h = C_L/C_{in}(nand2)$. We deduce the input capacitance of the matched NAND gate given that we recall logical effort $g_{nand2} = C_{in}(nand2)/C_{inv} = 4/3$. Rearranging the definition of

g_{nand2}, we find $C_{in}(nand2) = g_{nand2}C_{inv} = 4/3 \cdot 3 = 4$ capacitive units. Now, for given load $C_L = 18$ we obtain $h = 18/4 = 9/2$, and the delay of the matched 2-input NAND gate is

$$d_{\gamma=1} = \frac{4}{3} \cdot \frac{18}{4} + 2 = 8$$

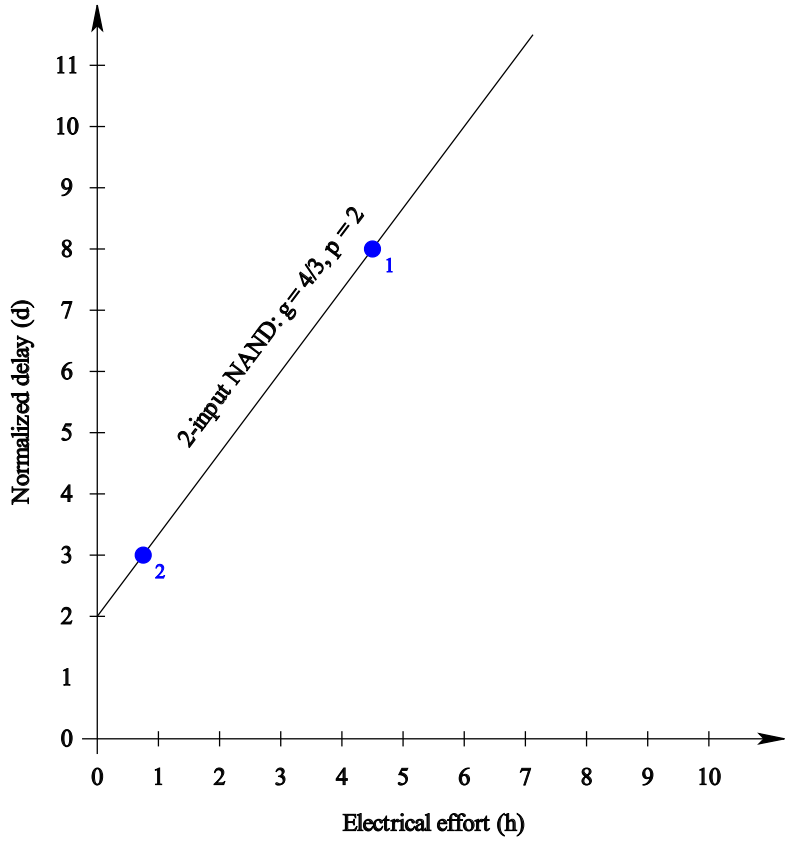
:ime units.

Scaling the NAND gate with scale factor $\gamma = 6$ yields a normalized input capacitance of $C_{in}(nand2) = \gamma \cdot 4 = 24$ capacitive units. Correspondingly, the electrical effort decreases by a factor of 6 to $h = C_L/C_{in}(nand2) = 18/24 = 3/4$. Since scaling affects neither the logical effort nor the parasitic delay of a gate, the delay of the scaled NAND gate driving load $C_L = 18$ is

$$d_{\gamma=6} = \frac{4}{3} \cdot \frac{3}{4} + 2 = 3$$

:ime units, which is almost three times faster than the matched NAND gate.

The $d(h)$-diagram below shows delay point 1, $(d, h) = (9/2, 8)$, and delay point 2, $(3/4, 3)$. Both points lie on the delay line of the 2-input NAND gate.



Hide

3.2. Delay of Two-Stage Paths

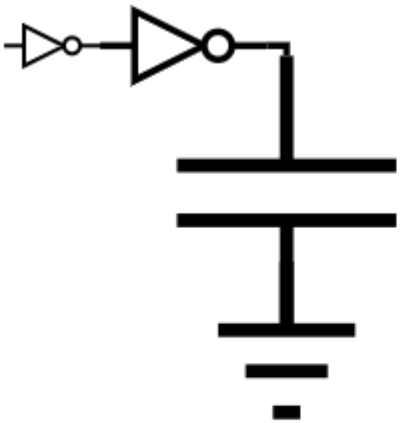
In this section we study the delay of a two-stage path with two gates in series. **Experiment 3.3** examines an inverter pair.

▶ **Experiment 3.3:** Delay of Inverter Pair

Consider the path below with back-to-back inverters driving a load capacitance. You can vary the size of the inverters and the load with the sliders below the schematics, and inspect the associated circuit delay.

1. Vary the size of the stage-2 inverter. Notice that the delay increases towards both small and large sizes. The minimum delay of 8.33 units occurs at scale factor $\gamma_2 = 3$ for inverter 2, assuming inverter 1 is sized like a reference inverter, $\gamma_1 = 1$, and the load is $C_L = 30$ capacitive units.
2. Double the size of inverter 1, $\gamma_1 = 2$, and vary the size of inverter 2. The qualitative behavior is the same as in experiment 1: The minimum delay of 6.5 units occurs at scale factors $\gamma_2 = 4$ and $\gamma_2 = 5$ for inverter 2.
3. Increase the load to $C_L = 60$ capacitive units, and vary the size of inverter 2. The qualitative behavior is the same as in experiment 1. The minimum delay is 8.33 units, if inverter 1 is scaled by $\gamma_2 = 2$.
4. Increase the size of inverter 1 to scale factor $\gamma_1 = 10$. Now, the delay decreases monotonically with increasing size of inverter 2. The minimum delay with a load of size 60 is 5 time units.

We observe that the delay is sensitive to the size of inverter 2. There exists a minimum delay, if the size of inverter 2 is chosen carefully, neither too small nor too large.



Inverter 1: 1 Inverter 2: 3

Load: 30 Delay: 8.33

In the following we study the delay of the generic path in **Figure 3.1** with two gates in series, gate 1 and gate 2, and gate 2 drives capacitive load C_L. Gate 1 has logical effort g_1, parasitic delay p_1, and is scaled such that its input capacitance is C_1. Analogously, gate 2 has logical effort g_2, parasitic delay p_2, and input capacitance C_2.

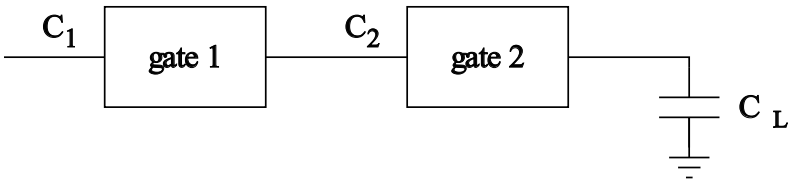


Figure 3.1: Generic two-stage path with two gates in series and capacitive load C_L.

The delay of the path is the sum of the delays of each gate. The input capacitance of gate 2 is the load capacitance of gate 1. Thus, gate 1 has electrical effort $h_1 = C_2/C_1$ and delay

$$d_1 = f_1 + p_1 = g_1 h_1 + p_1 = g_1 \frac{C_2}{C_1} + p_1 \,.$$

Gate 2 has electrical effort $h_2 = C_L / C_2$ and delay

$$d_2 = f_2 + p_2 = g_2 h_2 + p_2 = g_2 \frac{C_L}{C_2} + p_2 \,.$$

The **path delay** is the sum of the gate delays:

$$D = d_1 + d_2 = \left(g_1 \frac{C_2}{C_1} + p_1 \right) + \left(g_2 \frac{C_L}{C_2} + p_2 \right) .$$

Writing path delay D in terms of stage efforts f_1 and f_2 yields

$$D = (f_1 + f_2) + (p_1 + p_2) = F + P \,,$$

which has the same form as the delay of a single gate, i.e. the path delay is the sum of the **path effort F** and **path parasitic delay P**.

We wish to minimize path delay D, assuming that we can resize gate 2. If the scale factor for gate 2 is our only design parameter, then input capacitance C_2 is the only free variable in D. In particular, since parasitic delay P remains constant under sizing, minimizing D requires minimizing path effort $F = f_1 + f_2$.

Notice that the product of the stage efforts

$$f_1 \cdot f_2 = \left(g_1 \frac{C_2}{C_1} \right) \cdot \left(g_2 \frac{C_L}{C_2} \right) = g_1 g_2 \frac{C_L}{C_1}$$

is independent of C_2. Therefore, the geometric mean of the stage efforts $\sqrt{f_1 \, f_2}$ remains constant under changes of C_2. Since the logical efforts and capacitances of a circuit are positive numbers, we can apply the ***Theorem of Arithmetic and Geometric Means*** to minimize the path effort as follows. The geometric mean of the stage efforts cannot be greater than their arithmetic mean

$$\frac{f_1 + f_2}{2} \geq \sqrt{f_1 \, f_2} \,,$$

and equality holds if and only if the stage efforts are equal, $f_1 = f_2$. Furthermore, according to the ***corollary of the minimium arithmetic mean***, the arithmetic mean assumes its minimum for equal stage efforts, if the geometric mean is constant. Therefore, we conclude that **the path effort is minimal if the stage efforts are equal**:

$$f_1 = f_2 \quad \Rightarrow \quad \min F = f_1 + f_2 = 2\sqrt{f_1\, f_2}\,.$$

This is the crucial insight that enables us to minimize the path delay by sizing gate 2. Gate 2 must have an input capacitance C_2 that yields equal stage efforts:

$$\begin{aligned} f_1 &= f_2 \\ g_1 \frac{C_2}{C_1} &= g_2 \frac{C_L}{C_2} \\ C_2 &= \sqrt{\frac{g_2}{g_1} C_1 C_L}\,. \end{aligned}$$

With this choice of C_2, let $\hat{f}$ be the **stage effort that minimizes the path effort**, i.e.

$$\hat{f} = f_1 = f_2 = \sqrt{g_1 g_2 \frac{C_L}{C_1}}\,.$$

Then, we can write the minimal path effort as:

$$\min F = 2\hat{f} = 2\sqrt{g_1 g_2 \frac{C_L}{C_1}}\,.$$

We introduce the **path logical effort** $G = g_1 g_2$ and the **path electrical effort** $H = h_1 h_2 = C_L/C_1$, such that $\hat{f} = \sqrt{GH}$. Then, we can write the minimum path delay $\hat{D} = 2\hat{f} + P$ of the two-stage path as:

$$\hat{D} = 2\sqrt{GH} + P\,.$$

Path logical effort G and path electrical effort H are easily identified in a given circuit. Path logical effort $G = g_1 g_2$ is the product of the logical efforts of the gates on the path. Path electrical effort $H = C_L/C_1$ is the fanout of the path, i.e. the ratio of the load capacitance and the input capacitance of the path. The path electrical effort is also the product of the stage electrical efforts

$$H = h_1 h_2 = \frac{C_2}{C_1} \cdot \frac{C_L}{C_2} = \frac{C_L}{C_1}\,.$$

Capacitance C_2 serves as both input and load capacitance and cancels out. Thus, to minimize the path delay of a two-stage path, all we need to do is identify G and H, because $\hat{f} = \sqrt{GH}$.

In summary, given a two-stage circuit with load capacitance C_L and gate 1 with input capacitance C_1, we may size gate 2 to minimize the path delay. Stage effort $\hat{f} = \sqrt{GH}$ determines input capacitance C_2 of gate 2.

If we wish to know the scale factor for gate 2, we recall two facts from the ***Model of Logical Effort***: (1) the logical effort of a gate is the ratio of the input capacitance of the ***matched gate*** and the reference inverter with input capacitance $C_{inv} = 3$:

$$g_{gate} = \frac{C_{in}(\text{matched gate})}{C_{inv}},$$

and (2) the input capacitance of a sized gate is scale factor γ times the input capacitance of the matched gate:

$$C_{in}(\text{gate}) = \gamma C_{in}(\text{matched gate}).$$

These basic facts determine scale factor γ:

$$\gamma = \frac{C_{in}(\text{gate})}{3g_{gate}}.$$

In case of gate 2 of the two-stage path, given C_2 we obtain scale factor γ_2:

$$\gamma_2 = \frac{C_2}{3g_2}.$$

The following examples illustrate the design of two-stage paths.

■ **Example 3.1:** Minimize Delay of Inverter Pair

We analyze the path delay of the inverter pair in **Experiment 3.3**. In terms of the generic two-stage path in **Figure 3.1** both gates are inverters with identical logical efforts, $g_1 = g_2 = g_{inv} = 1$, and identical parasitic delays, $p_1 = p_2 = p_{inv} = 1$. The input capacitance of a sized inverter is $C_{in} = \gamma C_{inv} = 3\gamma$. The scale factors of the inverters, γ_1 and γ_2, are determined by the position of the sliders in **Experiment 3.3**. Therefore, the input capacitances are $C_1 = 3\gamma_1$ and $C_2 = 3\gamma_2$. Furthermore, load capacitance C_L can be adjusted with the slider in **Experiment 3.3**.

Given input capacitance C_1 and load capacitance C_L, we wish to minimize the delay of the path. Path delay D is minimal if the stage efforts are equal to $\hat{f} = \sqrt{GH}$. Path logical effort G is the product of the logical efforts of the inverters:

$$G = g_1 g_2 = 1 \cdot 1 = 1.$$

Path electrical effort H is the ratio of load capacitance C_L and path input capacitance C_1:

$$H = \frac{C_L}{C_1} = \frac{C_L}{3\gamma_1}\,.$$

We conclude that

$$\hat{f} = \sqrt{GH} = \sqrt{\frac{C_L}{3\gamma_1}}\,.$$

Thus, we minimize the path delay by choosing C_2 such that both stage efforts are equal to $\hat{f}$. We have several equations available to calculate C_2. First, there is

$$C_2 = \sqrt{\frac{g_2}{g_1} C_1 C_L} = \sqrt{3\gamma_1 C_L}\,.$$

Second, we may use the fact that stage effort f_2 must be equal to $\hat{f}$:

$$\begin{aligned} f_2 &\stackrel{!}{=} \hat{f} \\ g_2 \frac{C_L}{C_2} &= \sqrt{GH} \\ C_2 &= \sqrt{3\gamma_1 C_L}\,, \end{aligned}$$

and, third, stage effort f_1 must be equal to $\hat{f}$. All three equations yield the same result for C_2. With this choice of C_2, and path parasitic delay $P = p_1 + p_2 = 2$, the minimum path delay is

$$\hat{D} = 2\sqrt{GH} + P = 2\sqrt{\frac{C_L}{3\gamma_1}} + 2\,.$$

Hence, given γ_1 and C_L, we minimize the path delay by adjusting scale factor γ_2 such that

$$\gamma_2 = \frac{C_2}{3} = \sqrt{\frac{\gamma_1 C_L}{3}}\,.$$

This knowledge takes the guesswork out of **Experiment 3.3**. Instead, we can predict the delay and design the circuit for minimum delay. For example, given $\gamma_1 = 1$ and $C_L = 48$, we choose $\gamma_2 = 4$ to obtain minimum path delay $\hat{D} = 10$. As another example, consider $\gamma_1 = 10$ and $C_L = 60$. We would obtain minimum path delay

$\hat{D} \approx 4.83$ with $\gamma_2 = \sqrt{200} \approx 14.14$. Thus, the minimum path delay lies outside the slider range for inverter 2.

■ **Example 3.2:** Delay of Inverter Pair

Consider **Experiment 3.3** with fixed load $C_L = 21$. Then, the path delay is:

$$\begin{aligned} D &= \left(g_1 \frac{C_2}{C_1} + p_1\right) + \left(g_2 \frac{C_L}{C_2} + p_2\right) \\ &= \left(1\frac{3\gamma_2}{3\gamma_1} + 1\right) + \left(1\frac{21}{3\gamma_2} + 1\right) \\ &= \frac{\gamma_2}{\gamma_1} + \frac{7}{\gamma_2} + 2 \end{aligned}$$

If both inverters are reference inverters, i.e. $\gamma_1 = \gamma_2 = 1$, then the stage efforts of the inverters are

$$f_1 = \frac{\gamma_2}{\gamma_1} = 1\,, \qquad f_2 = \frac{7}{\gamma_2} = 7\,,$$

and the path delay is

$$D = 1 + 7 + 2 = 10\,.$$

We find that the stage efforts are unbalanced, with the second inverter bearing seven times the effort of the first inverter. If we increase the size of the second inverter to bear stage effort $f_2 = 1$, then scale factor γ_2 must be $\gamma_2 = 7/f_2 = 7$. Since C_2 is the load capacitance of the first inverter, changing γ_2 affects stage effort f_1. Thus, resizing the second inverter to have $\gamma_2 = 7$ results in stage efforts

$$f_1 = \frac{\gamma_2}{\gamma_1} = 7\,, \qquad f_2 = \frac{7}{\gamma_2} = 1\,,$$

and path delay

$$D = 7 + 1 + 2 = 10\,.$$

The delay is the same, but the burden of the larger stage effort has shifted from stage 2 to stage 1. If we keep $\gamma_1 = 1$ constant, then we can express path delay D as a function of scale factor γ_2:

$$D(\gamma_2) = \gamma_2 + \frac{7}{\gamma_2} + 2\,.$$

The plot $D(\gamma_2)$ in **Figure 3.2** shows that the path delay assumes its minimum if the stage efforts are equal, i.e. for $\gamma_2 = \sqrt{7}$.

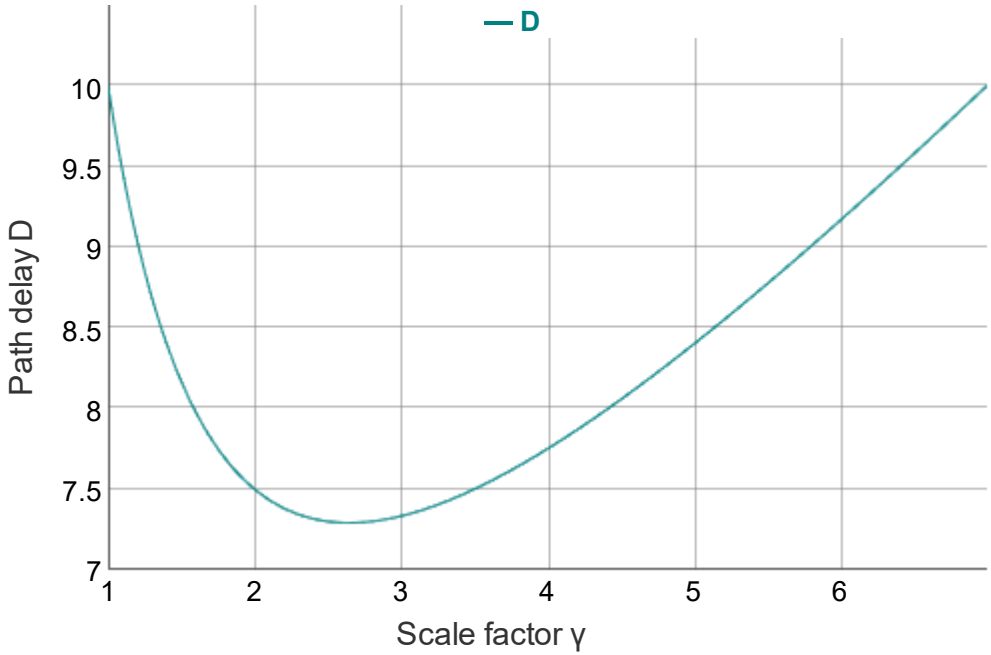
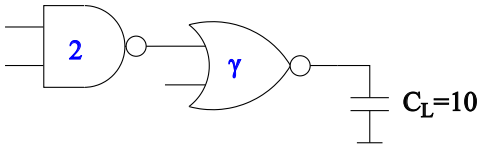


Figure 3.2: Path delay of inverter pair as a function of scale factor γ_2 of the inverter in stage 2. The inverter in stage 1 is a reference inverter and path electrical effort $H = 7$.

Minimize the delay of the 2-stage path using calculus.



a. Derive the stage delay for each stage.
b. Express path delay D as a function of scale factor γ and minimize $D(\gamma)$ using ***calculus***.
c. Compute minimum path delay $\hat{D}$ and the associated scale factor γ for the NOR gate.

The 2-stage path has a 2-input NAND gate in stage 1 and a 2-input NOR gate in stage 2. According to the ***Model of Logical Effort***, the NAND gate incurs delay

$$d_1 = g_{nand2} h_1 + p_{nand2} = \frac{4}{3} \frac{C_{in}(nor2)}{C_{in}(nand2)} + 2,$$

and the NOR gate has delay

$$d_2 = g_{nor2} h_2 + p_{nor2} = \frac{5}{3} \frac{C_L}{C_{in}(nor2)} + 2.$$

The normalized input capacitances depend on the scale factors. The NAND gate is scaled by factor 2, such that $C_{in}(nand2) = 2 \cdot 4 = 8$, because the normalized input capacitance of the matched 2-input NAND gate is 4. Scale factor γ of the NOR gate is variable. Since the matched 2-input NOR gate has a normalized input capacitance of 5 units, we have $C_{in}(nor2) = 5\gamma$. Given load capacitance $C_L = 10$, the stage delays are

$$d_1 = \frac{4}{3} \frac{5\gamma}{8} + 2 = \frac{5\gamma}{6} + 2$$

and

$$d_2 = \frac{5}{3} \frac{10}{5\gamma} + 2 = \frac{10}{3\gamma} + 2.$$

The path delay is the sum of the stage delays:

$$D(\gamma) = d_1 + d_2 = \frac{5\gamma}{6} + \frac{10}{3\gamma} + 4,$$

and is a function of γ. ***Calculus*** tells us that function $D(\gamma)$ has a minimum if the derivative is zero. The derivative of path delay $D(\gamma)$ is

$$\frac{dD}{d\gamma} = \frac{5}{6} - \frac{10}{3\gamma^2}.$$

Setting the derivative to zero yields γ:

$$\begin{aligned} \frac{5}{6} - \frac{10}{3\gamma^2} &= 0 \\ \gamma^2 &= 4. \end{aligned}$$

Since the scale factor for a CMOS gate must be positive, we find $\gamma = 2$. The minimum path delay is

$$\hat{D} = D(2) = \frac{5 \cdot 2}{6} + \frac{10}{3 \cdot 2} + 4 = \frac{22}{3}$$

:ime units. Strictly speaking, we should also inspect the second derivative to be sure that $\gamma = 2$ yields a minimum rather than a maximum.

As circuit designers with a choice to size the NOR gate, we minimize the delay of the 2-stage path by scaling the NOR gate with factor $\gamma = 2$, and obtain a minimum path delay of $\hat{D} = 22/3$ time units.

Hide

Use the method of logical effort to minimize the delay of the 2-stage path in **Exercise 3.4**.

a. Determine path logical effort G, path electrical effort H, and path parasitic delay P.
b. Calculate best stage effort $\hat{f} = \sqrt{GH}$.
c. Calculate minimum path delay $\hat{D} = 2\hat{f} + P$ and the associated scale factor γ of the NOR gate.

We begin by identifying the characteristic metrics of the 2-stage path. Path logical effort G is the product of the gate logical efforts:

$$G = g_{nand2}\, g_{nor2} = \frac{4}{3}\,\frac{5}{3} = \frac{20}{9}.$$

Path electrical effort H is the ratio of the load capacitance of the path and the path input capacitance:

$$H = \frac{C_L}{C_{in}(nand2)} = \frac{10}{2 \cdot 4} = \frac{5}{4}.$$

Path parasitic delay P is the sum of the gate parasitic delays:

$$P = p_{nand2} + p_{nor2} = 2 + 2 = 4.$$

The best stage effort that minimizes the path delay is

$$\hat{f} = \sqrt{GH} = \sqrt{\frac{20}{9}\,\frac{5}{4}} = \frac{5}{3}.$$

When each stage of the 2-stage path bears stage effort $\hat{f}$ the minimum path delay is

$$\hat{D} = 2\hat{f} + P = 2\frac{5}{3} + 4 = \frac{22}{3}.$$

To determine scale factor γ of the NOR gate we exploit the fact that the normalized input capacitance of the scaled NOR gate is $C_2 = \gamma C_{in}(nor2)$, where $C_{in}(nor2) = 5$ is the input capacitance of the matched NOR gate, and that best stage effort $\hat{f} = g_2 h_2$:

$$\begin{aligned}\hat{f} &= g_{nor2}\frac{C_L}{5\gamma}\\ \gamma &= g_{nor2}\frac{C_L}{5\hat{f}}\\ &= \frac{5}{3}\frac{10}{5\cdot 5/3}\\ &= 2.\end{aligned}$$

Alternatively, we could exploit the fact that stage 1 of the circuit, the NAND gate, must bear the best stage effort as well, i.e. $\hat{f} = g_1 h_1$ to minimize the path delay.

Hide

3.3. Multistage Paths

An n-stage path provides more than just a straightforward generalization of the two-stage path to $n \geq 1$ stages. The number of stages of a path can serve as additional design parameter for a circuit, because the delay of a path is not only a function of the gate sizes but is also a function of the number of stages. In this section we study the path sizing problem of how to choose the number of stages together with the gate sizing problem in order to minimize its delay.

3.3.1. Gate Sizing

The **gate sizing problem** determines the scale factors for the gates on an n-stage path in order to minimize its delay. **Figure 3.3** shows a generic path with n gates in series and load capacitance C_L. In stage i, $1 \leq i \leq n$, logic gate i has input capacitance C_i, logical effort g_i, electrical effort h_i, and parasitic delay p_i.

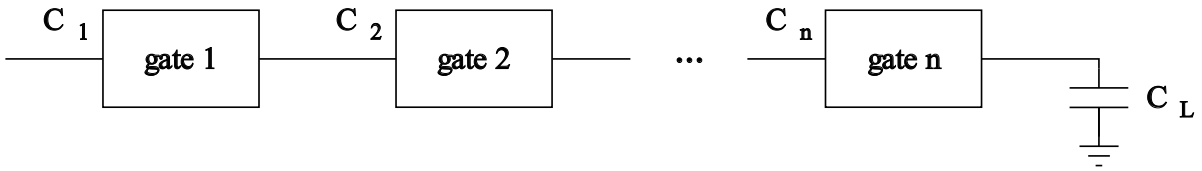


Figure 3.3: Generic n-stage path with n gates in series and capacitive load C_L.

The path delay of the n-stage path is the sum of the stage delays d_i:

$$
\begin{aligned}
D &= \sum_{i=1}^{n} d_i \\
&= \sum_{i=1}^{n} (g_i h_i + p_i) \\
&= \sum_{i=1}^{n} g_i h_i \; + \; \sum_{i=1}^{n} p_i \, .
\end{aligned}
$$

The path logical effort of the n-stage path is

$$
G = \prod_{i=1}^{n} g_i \, ,
$$

the path electrical effort is

$$
H = \prod_{i=1}^{n} h_i = \frac{C_2}{C_1} \cdot \frac{C_3}{C_2} \cdots \frac{C_L}{C_n} = \frac{C_L}{C_1} \, ,
$$

and the path parasitic delay is

$$
P = \sum_{i=1}^{n} p_i \, .
$$

Our goal is to minimize path delay $D = \sum_{i=1}^{n} (g_i h_i) + P$ under the constraint of a constant path electrical effort $H = C_L / C_1$. Observe that constant H implies that path effort $F = GH$ is constant, because all logical efforts g_i and, hence, G are constant. We invoke the ***Theorem of Arithmetic and Geometric Means*** to minimize path effort $F = \sum_{i=1}^{n} f_i = \sum_{i=1}^{n} g_i h_i$. Given a constant geometric mean $\sqrt[n]{GH} = \sqrt[n]{f_1 f_2 \cdots f_n}$, the arithmetic mean of the stage efforts is minimized, and is equal to the geometric mean if all stage efforts are equal:

$$
f_1 = f_2 = \cdots = f_n \quad \Rightarrow \quad \frac{f_1 + f_2 + \cdots + f_n}{n} = \sqrt[n]{f_1 f_2 \cdots f_n} \, .
$$

Therefore, the minimum path effort is

$$
\min F = f_1 + f_2 + \cdots + f_n = n \sqrt[n]{f_1 f_2 \cdots f_n} \, .
$$

Since path parasitic delay P is constant for a given path, we conclude that the path delay of an n-stage path assumes its minimum

$$\hat{D} = n\sqrt[n]{GH} + P = nF^{\frac{1}{n}} + P \tag{2}$$

if and only if all stages bear the same effort $\hat{f}$, equal to the n^{th} root of path effort F:

$$\hat{f} = g_1 h_1 = g_2 h_2 = \cdots = g_n h_n = F^{\frac{1}{n}} .$$

■ **Example 3.3:** Delay of 4-Inverter Path

We perform gate sizing to minimize the delay of the inverter chain in **Experiment 3.1**. The path consists of $n = 4$ inverters, and the stage-1 inverter is a reference inverter, that is $C_1 = C_{inv} = 3$.

Given load C_L, the path electrical effort of the inverter chain is $H = C_L/C_1 = C_L/3$. The path logical effort is $G = 1$, because the logical effort of an inverter is $g_{inv} = 1$ and $G = g_{inv}^4$. Furthermore, since the parasitic delay of an inverter is $p_{inv} = 1$, the path parasitic delay is $P = 4p_{inv} = 4$. Thus, the path delay of the 4-stage path is:

$$D = h_1 + h_2 + h_3 + h_4 + 4 .$$

To minimize the path delay, all stage efforts must be equal. Since all gates on the path are inverters with $g_{inv} = 1$, the stage effort equals the electrical effort $f_i = g_i h_i = h_i$. Thus, all stage electrical efforts must be equal, and must be equal to

$$F^{\frac{1}{4}} = H^{\frac{1}{4}} = \left(\frac{C_L}{3}\right)^{\frac{1}{4}} .$$

This criterion enables us to determine the scale factors for each stage either by working from front to tail of the chain or vice versa. Let's work backwards starting with stage 4:

$$\begin{aligned} h_4 = \frac{C_L}{C_4} &\stackrel{!}{=} \left(\frac{C_L}{3}\right)^{\frac{1}{4}} \\ C_4 &= \left(3C_L^3\right)^{\frac{1}{4}} . \end{aligned}$$

The scale factor for the stage 4 inverter derives from $C_4 = \gamma_4 C_{inv}$ as

$$\gamma_4 = \frac{C_4}{3} = \left(\frac{C_L}{3}\right)^{\frac{3}{4}} = \left(\sqrt[4]{C_L/3}\right)^3 .$$

Knowing C_4, we can size stage 3:

$$h_3 = \frac{C_4}{C_3} \stackrel{!}{=} \left(\frac{C_L}{3}\right)^{\frac{1}{4}}$$
$$C_3 = \left(9C_L^2\right)^{\frac{1}{4}}$$

and the corresponding scale factor is:

$$\gamma_3 = \frac{C_3}{3} = \left(\frac{C_L}{3}\right)^{\frac{1}{2}} = \left(\sqrt[4]{C_L/3}\right)^2 .$$

Analogously, we obtain for stage 2:

$$h_2 = \frac{C_3}{C_2} \stackrel{!}{=} \left(\frac{C_L}{3}\right)^{\frac{1}{4}}$$
$$C_2 = \left(27C_L\right)^{\frac{1}{4}}$$

with scale factor

$$\gamma_2 = \frac{C_2}{3} = \left(\frac{C_L}{3}\right)^{\frac{1}{4}} = \left(\sqrt[4]{C_L/3}\right)^1 .$$

The stage-1 inverter is a reference inverter with scale factor is $\gamma_1 = 1.$

Since $C_L/3 = H,$ we can list the scale factors emphasizing their common form:

$$\gamma_1 = \sqrt[4]{H}^0 , \quad \gamma_2 = \sqrt[4]{H}^1 , \quad \gamma_3 = \sqrt[4]{H}^2 , \quad \gamma_4 = \sqrt[4]{H}^3 ,$$

and see that the scale factor for stage i for $i > 0$ is

$$\gamma_i = \sqrt[4]{H}^{i-1} .$$

The scale factors form a geometric progression. This observation generalizes to arbitrary n-stage inverter chains. We conclude that **minimizing the path delay of an inverter chain requires that all inverters must bear equal stage effort, which we accomplish by growing the inverter sizes geometrically**.

With the inverter sizes determined above, the minimum path delay of the 4-stage path is according to Equation **(2)**:

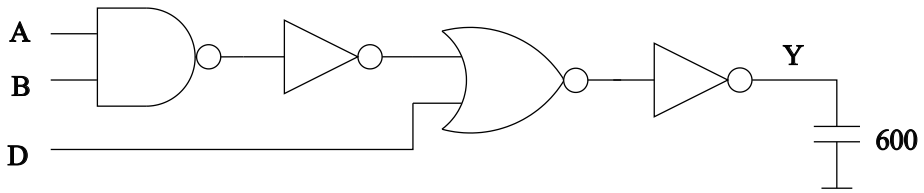
$$\hat{D} = 4F^{\frac{1}{4}} + P = 4\left(\frac{C_L}{3}\right)^{\frac{1}{4}} + 4 .$$

For example, given $C_L = 1200,$ we obtain minimum delay $\hat{D} = 21.9$ by scaling the inverters with factors $\gamma_2 = \sqrt[4]{400}^1 = 4.47,$ $\gamma_3 = \sqrt[4]{400}^2 = 20,$ and $\gamma_4 = \sqrt[4]{400}^3 = 89.4.$ The sliders in **Experiment 3.1** do not support fractions. Thus, we may round the scale factors to $\gamma_2 = 4,$ $\gamma_3 = 20,$ and $\gamma_4 = 89.$ To calculate the resulting path delay, we resort to the sum of the stage delays:

$$\begin{aligned} D &= h_1 + h_2 + h_3 + h_4 + P \\ &= \frac{\gamma_2}{1} + \frac{\gamma_3}{\gamma_2} + \frac{\gamma_4}{\gamma_3} + \frac{1200}{3\gamma_4} + 4 \\ &= 4 + 5 + 4.45 + 4.49 + 4 \;=\; 21.94\,. \end{aligned}$$

We note that rounding causes unequal stage efforts in range $4 \le f_i \le 5.$ However, the differences are small enough to slow down the circuit by $0.18\,\%$ only.

■ **Example 3.4:** Delay of Logic Path



Consider a circuit for $Y = AB + D.$ The AND gate is implemented with a NAND gate followed by an inverter, and the OR gate is implemented with a NOR gate followed by an inverter. The NAND gate shall be sized like a matched NAND gate and the load driven by output Y shall be $C_L = 600$ capacitive units. Perform gate sizing to minimize the delay of the path.

We observe that the paths $A \to Y$ and $B \to Y$ are equal, whereas path $D \to Y$ passes through the NOR gate and the NOR inverter only. Thus, we assume that the **path of interest** is $A \to Y,$ because it incurs the largest delay. Path $B \to Y$ has the same delay, and the delay of $D \to Y$ is smaller.

Our plan is to determine the minimum path delay, and then derive the gate sizes from the associated stage effort. To simplify the notation, we enumerate the gates on the path: the NAND gate is gate 1, the inverter gate 2, the NOR gate is gate 3, and the second inverter is gate 4. The path consists of $n = 4$ gates. Therefore, the minimum path delay is

$$\hat{D} = 4F^{\frac{1}{4}} + P\,.$$

To determine path effort $F = GH,$ we need to determine path logical effort G and path electrical effort $H.$ The path electrical effort is the ratio of load capacitance C_L and input capacitance $C_{in}(A).$ The input capacitance of the matched NAND gate is $C_1 = C_{in}(A) = 4$ units, cf. **Figure 1.48**. Thus, we find

$$H = \frac{C_L}{C_1} = \frac{600}{4} = 150\,.$$

The path logical effort is the product of the gate logical efforts:

$$\begin{aligned} G &= g_1 \cdot g_2 \cdot g_3 \cdot g_4 \\ &= g_{nand} \cdot g_{inv} \cdot g_{nor} \cdot g_{inv} \\ &= \frac{4}{3} \cdot 1 \cdot \frac{5}{3} \cdot 1 \\ &= \frac{20}{9} \,. \end{aligned}$$

Thus, the path effort is $F = GH = 1000/3 = 333.3.$ Together with path parasitic delay

$$\begin{aligned} P &= p_1 + p_2 + p_3 + p_4 \\ &= p_{nand} + p_{inv} + p_{nor} + p_{inv} \\ &= 2 + 1 + 2 + 1 \\ &= 6 \,, \end{aligned}$$

we obtain a minimum path delay of $\hat{D} = 4 \cdot 333.3^{1/4} + 6 = 23.1$ units.

To achieve minimum path delay, all four stages must bear the same stage effort $\hat{f} = F^{1/4} = 4.27.$ This constraint enables us to determine the gate sizes. The NAND gate in the first stage is a matched gate with scale factor $\gamma_1 = 1,$ as required by the problem formulation. The stage effort of the NAND gate is

$$f_1 = g_{nand} h_1 = \frac{4}{3} \frac{C_2}{C_1} = \frac{C_2}{3} \,,$$

because the matched NAND gate has input capacitance $C_1 = 4.$ For minimum path delay, the stage effort must be $f_1 = \hat{f},$ such that $C_2 = 3\,\hat{f}.$ Thus, the scale factor for the stage-2 inverter must be

$$\gamma_2 = \frac{C_2}{C_{inv}} = \frac{C_2}{3} = \hat{f} = 4.27 \,,$$

because the reference inverter has input capacitance $C_{inv} = 3.$ The stage effort of the inverter is

$$f_2 = g_{inv} h_2 = 1 \cdot \frac{C_3}{C_2} = \frac{C_3}{3\hat{f}} \,.$$

Since minimum path delay requires $f_2 = \hat{f},$ the input capacitance of the NOR gate is $C_3 = 3\hat{f}^2.$ Because the matched NOR gate has input capacitance $C_{in}(nor) = 3g_{nor},$ the scale factor of the NOR gate in stage 3 must be

$$\gamma_3 = \frac{C_3}{3g_{nor}} = \frac{3\hat{f}^2}{3 \cdot 5/3} = \frac{3}{5}4.27^2 = 10.94\,.$$

The stage effort of the NOR gate is

$$f_3 = g_{nor}h_3 = \frac{5}{3}\frac{C_4}{C_3} = \frac{5}{9\hat{f}^2}C_4\,.$$

Because the NOR gate must bear stage effort $f_3 = \hat{f}$, we obtain the input capacitance of the stage-4 inverter, $C_4 = (9/5)\hat{f}^3$. The scale factor of the stage-4 inverter must be

$$\gamma_4 = \frac{C_4}{3g_{inv}} = \frac{3}{5}\hat{f}^3 = 46.7\,.$$

Like all other gates, the stage-4 inverter must bear stage effort $\hat{f}$. Because $f_4 = g_{inv}h_4 = C_L/C_4$, we can check our arithmetic:

$$\begin{aligned} \frac{C_L}{C_4} &\stackrel{!}{=} \hat{f} \\ \Leftrightarrow \qquad C_L &= \frac{9}{5}\hat{f}^4 = \frac{9}{5}4.27^4 = 598.4\,, \end{aligned}$$

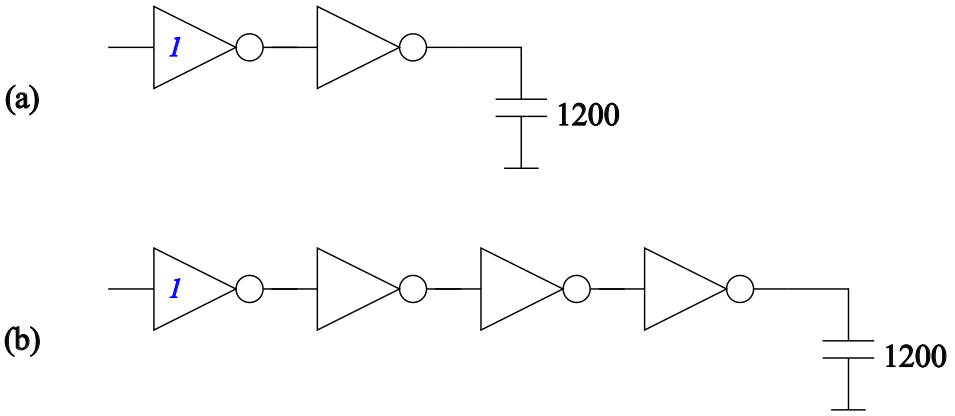
which is equal to 600 within the accuracy of the arithmetic precision. Had we used additional fractional digits for the value of $\hat{f}$, e.g. $\hat{f} = 4.273$, we would have obtained a more accurate arithmetic result.

Note that the scale factors do not form a geometric progression as for inverter chains with minimum path delay, see **Example 3.3**, because the logic path includes logical efforts. However, we may recognize the sequence as a geometric progression, weighted with logical efforts:

$$\gamma_1 = \hat{f}^0\,, \quad \gamma_2 = \hat{f}^1\,, \quad \gamma_3 = \frac{3}{5}\hat{f}^2\,, \quad \gamma_4 = \frac{3}{5}\hat{f}^3\,.$$

We would have arrived at the same scale factors by working backwards through the path, starting with the stage-4 inverter. We leave it as an exercise to double check that these scale factors yield the minimum path delay by first computing the individual stage delays $d_i = g_i h_i + p_i$, and then taking their sum $\hat{D} = \sum_{i=1}^4 d_i$.

■ **Example 3.5:** Inverter Path Length



We wish to drive a load capacitance of 1200 units, either with (a) a single buffer or (b) two back-to-back buffers. In both circuits the stage-1 inverter shall be a reference inverter. The remaining inverters may be sized for minimum delay.

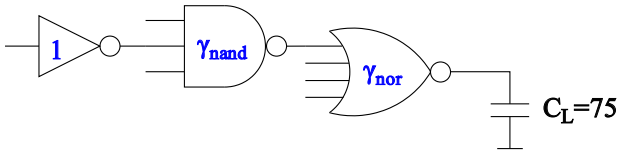
Intuitively, we may prefer the shorter 2-stage path of option (a), expecting that two inverters have a smaller path delay than four. This is not always the case, however.

In **Example 3.3** we minimize the delay of the 4-stage path in option (b). The smallest possible delay of the 4-inverter chain is $\hat{D}_4 = 21.9$ delay units. The inverter pair of option (a) constitutes a 2-stage path. According to Equation **(2)**, the minimum path delay with load $C_L = 1200$ and input capacitance $C_1 = 3,$ i.e. $F = H = C_L/C_1 = 400,$ is

$$\begin{aligned}\hat{D}_2 &= nF^{\frac{1}{n}} + n \\ &= 2\sqrt{400} + 2 \\ &= 42\end{aligned}$$

delay units. We find that the minimum delay of the 2-stage path $\hat{D}_2 = 42$ is almost two times larger than the minimum delay of the 4-stage path $\hat{D}_4 = 21.9.$ The next section on path sizing offers the background that puts this result into perspective.

Minimize the delay of the 3-stage path using the method of logical effort.



a. Compute minimum path delay $\hat{D}$.
b. Compute the scale factors γ_{nand} and γ_{nor}.

Solution

3.3.2. Path Sizing

The **path sizing problem** determines the best number of stages for a path in order to minimize its delay. As a concrete example, consider the inverter chain with n inverters and load C_L shown in **Figure 3.4**.

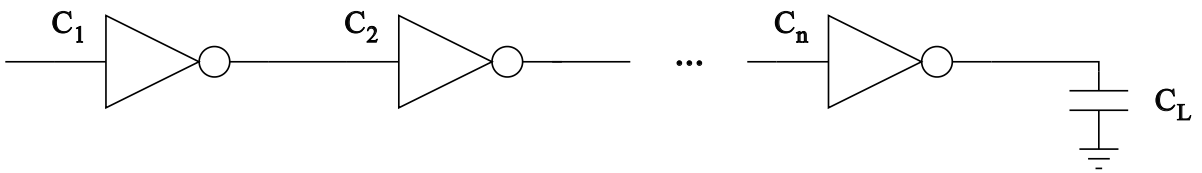


Figure 3.4: An n-stage inverter chain.

Given the number of stages n and path electrical effort $H = C_L/C_1$, we can minimize the path delay through gate sizing. According to Equation **(2)**, the minimum path delay of a chain with n inverters is

$$\hat{D} = nH^{\frac{1}{n}} + n\,.$$

Figure 3.5 plots the minimum path delay as a function of the number of stages n for fixed path electrical effort $H = 400.$ Note that the minimum path delay $\hat{D}(n)$ assumes the minimum $\hat{D}(n^\star) = 21.57$ for a number of stages $n^\star = 5.$ Whereas gate sizing enables us to find a *local minimum* of the path delay for a given number of stages n, finding the *global minimum* of the path delay requires optimizing the number of stages n.

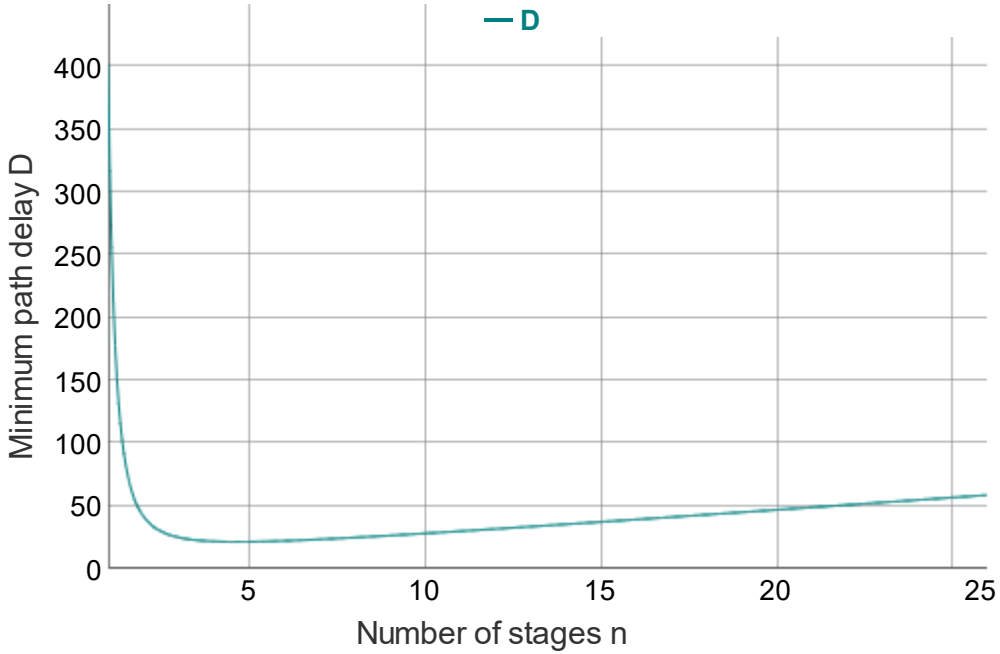


Figure 3.5: Minimum path delay of an n-stage inverter chain as a function of the number stages n for fixed path electrical effort $H = 400$.

Given path electrical effort H, we determine the best number of stages $n^\star$ by minimizing $\hat{D}(n)$. We determine the minimum using calculus by deriving $\hat{D}$ w.r.t. n for fixed H:

$$\begin{aligned}\frac{d\hat{D}}{dn} &= \frac{d}{dn}\left(nH^{\frac{1}{n}} + n\right) \\ &= n\frac{dH^{\frac{1}{n}}}{dn} + H^{\frac{1}{n}} + 1 \\ &= n\left(H^{\frac{1}{n}} \cdot \ln H \cdot \left(-\frac{1}{n^2}\right)\right) + H^{\frac{1}{n}} + 1 \\ &= -H^{\frac{1}{n}} \cdot \ln H^{\frac{1}{n}} + H^{\frac{1}{n}} + 1\,.\end{aligned}$$

Function $\hat{D}(n)$ has a minimum where its derivative is zero. Call the **best stage effort** $f^\star = H^{1/n^\star}$ the stage effort with path electrical effort H when using the best number of stages $n^\star$. This is the stage effort where $\hat{D}(n)$ is minimal, i.e. where derivative $d\hat{D}/dn$ equals zero:

$$f^\star(\ln f^\star - 1) - 1 = 0\,. \tag{3}$$

We have arrived at an equation for best stage effort $f^\star$ that has no closed-form solution. However, we can compute $f^\star$ numerically, for example with Newton's method for finding zeros. In case of Equation (3), we want to find x such that function $g(x) = x(\ln x - 1) - 1$ is zero. With derivative $g'(x) = \ln x$, we obtain the Newton iteration with iteration index i:

$$\begin{aligned}x_{i+1} &= x_i - \frac{g(x_i)}{g'(x_i)} \\ &= x_i - \frac{x_i \ln x_i - x_i - 1}{\ln x_i} \\ &= \frac{x_i + 1}{\ln x_i}\,.\end{aligned}$$

Iterating with a suitable starting value, e.g. $x_0 = 4$, produces result $x \approx 3.59$. Thus, Equation (3) holds for best stage effort

$$f^\star = 3.59\,, \tag{4}$$

which enables us to derive the best number of stages

(5)

$$n^\star = \frac{\ln H}{\ln f^\star} = \log_{f^\star} H = \log_{3.59} H\,,$$

because $H = f^{\star n^\star}$ by definition and $\ln H = n^\star \ln f^\star$.

This result is very useful. First, since each inverter on the path must bear best stage effort $f^\star = 3.59$, we know how to derive the scale factors for sizing each of the inverters in geometric progression. Second, given a path with electrical effort H, we find the best number of stages by taking the logarithm of H to base 3.59. **Table 3.1** lists path efforts and the corresponding best number of stages. The numbers are rounded for ease of reading. For more accurate numerical results, use the **path sizing calculator** below the table. **Table 3.1** is suited to look up the best number of stages for a given path effort F, which equals the path electrical effort H in an inverter chain with $G = 1$. For example, using one stage is best for path efforts F up to 5.8. For efforts in range $5.8 \le F \le 22.3$, a 2-stage design is best, and so on. Path effort $F = 400$ of **Example 3.5** lies between 300 and 1090, where $n^\star = 5$ stages yield the fastest inverter chain.

path effort	stages	delay	stage effort
0		1.0	
	1		0-5.8
5.8		6.8	
	2		2.4-4.7
22.3		11.4	
	3		2.8-4.4
82.2		16.0	
	4		3.0-4.2
300		20.6	
	5		3.1-4.1
1090		25.3	
	6		3.2-4.0
3920		29.8	
	7		3.3-3.9
14200		34.4	
	8		3.3-3.9
51000		39.0	
	9		3.3-3.9

path effort F	stages $n^\star$	delay $\hat{D}(n^\star)$	stage effort $\hat{f}$
184000		43.6	
	10		3.4-3.8
661000		48.2	

Table 3.1: Best number of stages $n^\star$ for n-stage inverter chain.

Path Sizing Calculator

Path effort F: 10.0
Best number of stages: 2
Stage effort: 3.16
Path delay (inverter chain): 8.32

Table 3.1 also includes the minimum path delay $\hat{D}(n^\star)$ for the given path efforts. For example, given $F = 300,$ a 4-stage path yields $\hat{D}(4) = 4\sqrt[4]{300} + 4 = 20.6,$ as does a 5-stage path, $\hat{D}(5) = 5\sqrt[5]{300} + 5 = 20.6.$ Furthermore, the right-most column lists the corresponding stage efforts $\hat{f} = F^{1/n^\star}.$ For example, consider 5-stage inverter chains, which yield the smallest delay for path efforts in range $300 \leq F \leq 1090.$ At the lower end, for $F = 300,$ the stage effort with minimum delay is $\hat{f} = \sqrt[5]{300} = 3.1$ delay units, and at the upper end for $F = 1090,$ the stage effort is $\hat{f} = \sqrt[5]{1090} = 4.1$ delay units. The best stage effort $f^\star = 3.59$ of a 5-stage inverter chain produces the minimum delay for path effort $F = f^{\star 5} = 596.$

■ **Example 3.6:** Inverter Chain Design

We design an inverter chain with input capacitance $C_{in} = 3$ and load capacitance $C_L = 500$ with minimum delay.

The path electrical effort of the chain is

$$H = \frac{C_L}{C_{in}} = \frac{500}{3} = 166.7\,.$$

Since the path logical effort of an inverter chain is $G = 1,$ the path effort is $F = GH = 166.7.$ The best number of stages is

$$n^\star = \log_{3.59} 166.7 = 4\,.$$

Stage effort

$$\hat{f} = F^{\frac{1}{n^\star}} = 166.7^{\frac{1}{4}} = 3.59$$

happens to be equal to best stage effort $f^\star$. The delay of the 4-stage path is

$$\hat{D} = n^\star \cdot \hat{f} + n^\star = 4 \cdot 3.59 + 4 = 18.4\,,$$

if the inverters are sized in geometric progression. The stage-1 inverter must be a reference inverter, because the path input capacitance $C_{in} = 3$ is given. Thus, the scale factor of the inverter in stage i is $\hat{f}^{i-1}$, such that:

$$\begin{aligned}
\gamma_1 &= 3.59^0 &&= 1 \\
\gamma_2 &= 3.59^1 &&= 3.59 \\
\gamma_3 &= 3.59^2 &&= 12.9 \\
\gamma_4 &= 3.59^3 &&= 46.3\,.
\end{aligned}$$

The **path sizing calculator** and **Experiment 3.1** enable you to verify this result.

■ **Example 3.7:** Inverter Chain Path Sizing

Determine the number of stages for an inverter chain with path electrical effort $H = 24$ in order to minimize the delay.

According to Equation **(5)**, the best number of stages is

$$n^\star = \log_{3.59} H = \log_{3.59} 24 = 2.4864\,,$$

which is a real number rather than an integer. Since the number of inverters must be an integer, we might decide to round 2.4864 to the nearest integer 2. However, **Table 3.1** tells us that we should use 3 stages. The reason is that the delay of a 3-stage path is less than the delay of a 2-stage path for $H = 24$:

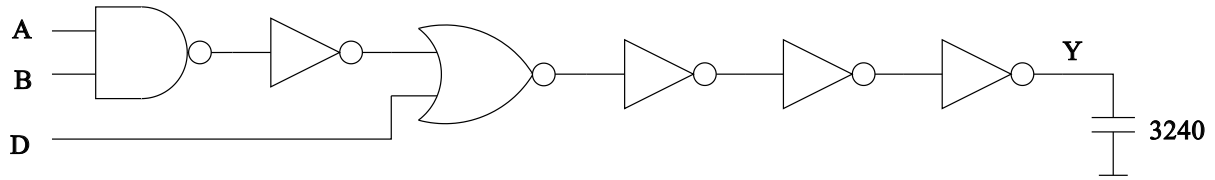
$$\begin{aligned}
\hat{D}(2) &= 2\sqrt{24} + 2 = 11.798\,, \\
\hat{D}(3) &= 3\sqrt[3]{24} + 3 = 11.653\,.
\end{aligned}$$

Although such a small difference between the delays rarely matters in practice, the lesson to be learned is that rounding $n^\star$ may not yield the best solution. Instead, to find the minimum delay you may want to check the delay for both numbers of stages the floor and the ceiling of $n^\star$.

■ **Example 3.8:** Path Sizing of Logic Circuit

Consider the circuit in **Example 3.4** with a larger capacitive load of $C_L = 3240$ units. We can increase the path length of a circuit by inserting buffers (inverter pairs) between its output and the load. Perform path sizing and determine the minimum path delay of the circuit.

Path $A \to Y$ of the circuit in **Example 3.4** has a matched NAND gate with input capacitance $C_1 = 4$ in stage 1. Therefore, the path must bear electrical effort $H = C_L/C_1 = 810.$ Since an inverter has logical effort $g_{inv} = 1,$ the circuit has path electrical effort $G = g_{nand} \cdot g_{nor} = 20/9,$ no matter how many inverters we insert on the path. Since the path electrical effort is also independent of the number of additional inverter stages, the extended path must bear path effort $F = GH = 1800.$ The **path sizing calculator** prescribes using 6 stages, or inserting two inverters at output $Y.$



The minimum delay of path $A \to Y$ is $\hat{D} = 6F^{1/6} + P = 6 \cdot 1800^{1/6} + 8 = 28.927,$ which exceeds the minimum delay of a 6-stage inverter chain, given by the path sizing calculator, by 2 parasitic delay units due to the NAND and NOR gates.

■ **Example 3.9:** Parasitic Delay of Inverter Chains

We investigate the contribution of parasitic delay to the total path delay of inverter chains.

Consider n-stage inverter chains with path efforts that are powers of the best stage effort:

$$F(n) = f^{\star n} = 3.59^n\,.$$

The corresponding minimum path delay is

$$\hat{D}(n) = nf^\star + n\,.$$

The second summand represents path parasitic delay $P = np_{inv} = n$ because $p_{inv} = 1.$ Thus, the percentage of parasitic delay of an n-stage inverter chain is

$$\frac{P}{\hat{D}(n)} = \frac{n}{n(f^\star + 1)} = \frac{1}{4.59} = 21.8\,\%\,,$$

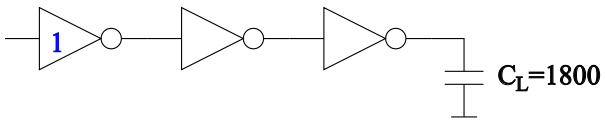
independent of the number of stages.

If the path effort differs from the powers of the best stage effort, yet we use the best number of stages according to **Table 3.1**, then the percentage of the path parasitic delay is largest at the lower end of the path effort range and smallest at the upper end. **Table 3.2** lists the percentages of path parasitic delay relative to the total delay for the $1 \leq n \leq 5$ for the path efforts at the boundaries of the ranges for which the number of stages is best. For example, $n = 3$ stages are best for $22.3 \leq F \leq 82.2$. At the lower end of the range, where $F = 22.3$, the path parasitic delay of 3 units contributes $26.3\,\%$ to the minimum total delay of 11.4 delay units. At the upper end, for $F = 82.2$, the parasitics amount to only $18.7\,\%$ of the the minimum total delay of 16.0 units.

n\F	0	5.8	22.3	82.2	300	1090
1	100	14.7				
2		29.3	17.5			
3			26.2	18.7		
4				24.9	19.4	
5					24.2	19.8

Table 3.2: Percentage of path parasitic delay.

Minimize the delay of the inverter chain by path sizing.



a. Compute $\hat{f}$ of the 3-stage inverter chain.
b. Determine the best number of stages $n^{\star}$.
c. Compute the minimum path delay $\hat{D}$ of the path-sized inverter chain.

The stage effort of the 3-stage inverter chain with a reference inverter in stage 1 and path electrical effort

$$H = \frac{C_L}{C_1} = \frac{1800}{3} = 600$$

s

$$\hat{f} = \sqrt[3]{H} = \sqrt[3]{600} = 8.43\,.$$

We observe that this stage effort exceeds the best stage effort $f^\star = 3.59$ by more than a factor of 2. This is a strong indication that more than three stages are needed to minimize the path delay. In fact, the best number of stages is

$$n^\star = \log_{3.59} H = \log_{3.59} 600 = 5\,.$$

Table 3.1 yields the same result. Using $n^\star = 5$ stages, the minimum path delay is

$$\hat{D}(5) = 5\,H^{1/5} + 5\,p_{inv} = 22.97$$

time units. For comparison, the minimum path delay with $n = 3$ stages would be $\hat{D}(3) = 3\,H^{1/3} + 3\,p_{inv} = 28.3$ time units.

Hide

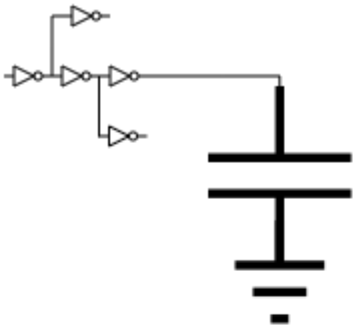
3.4. Branching Effort

So far, we have learned how to minimize the delay of a circuit path that drives a given load. We have assumed that the load is the equivalent load capacitance of one or more gates. Even if the load consists of a single logic gate only, the last stage of the path fans out to drive the transistor gate terminals of the logic gate, such as the nMOS and pMOS transistor gates in case of an inverter. In many circuits, fanout occurs not only at the end of a path, but anywhere along a path. When a gate in stage k on a path drives more than one gate in stage $k+1$, we say that the path **branches**. At the first glance, branching complicates the delay minimization of a circuit, as **Experiment 3.4** illustrates. However, the method of logical effort extends seamlessly to branching circuits.

▶ **Experiment 3.4:** Delay of Branching Path

Consider the 3-stage inverter chain below with branches after stage 1 and stage 2. The off-path inverters above and below the inverter chain are assumed to drive additional loads that are not shown, because they do not affect the delay of the chain.

The stage-1 inverter is a reference inverter. Choose the size of the off-path inverters and the load, and try sizing the on-path inverters with the goal to minimize the path delay of the 3-stage inverter chain.



Inverter 2: 1 Inverter 4: 1

Upper off-path inverter 3: 1 Lower off-path inverter 5: 1

Load: 80 Delay: 33.67

To understand the effect of branching on the delay of a circuit path, consider the annotated circuit in **Figure 3.6**. Our path of interest is $A \rightarrow Y$. We examine the branch at the output of stage-1 inverter 1 to off-path inverter 3.

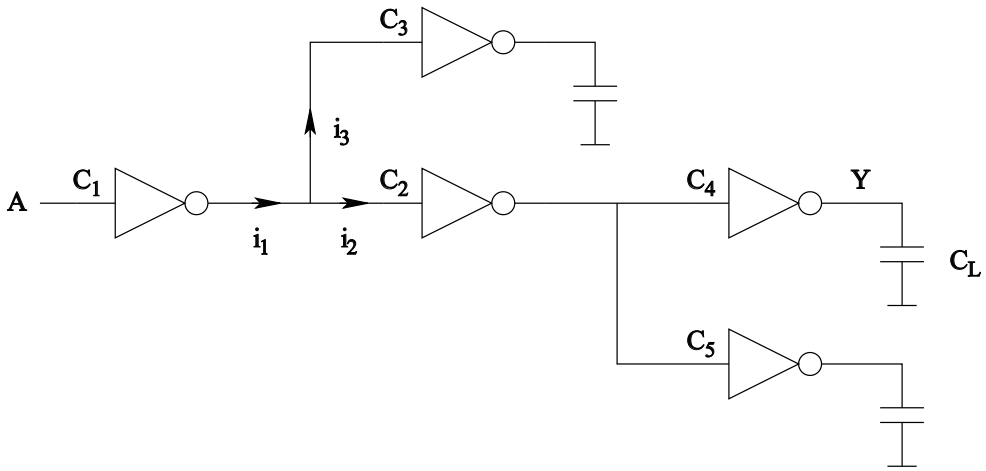


Figure 3.6: A branching circuit diverts drive current from the path of interest, $A \rightarrow Y$.

Assuming off-path inverter 3 were absent, then inverter 1 would invest its entire output current i_1 to drive inverter 2 on the path of interest. In contrast, in the presence of off-path inverter 3, current i_3 is diverted off the path of interest. Now, current i_2 drives on-path inverter 2, which, by ***KCL***, is:

$$i_2 = i_1 - i_3 \, .$$

Drive current i_2 is smaller than output current i_1 of inverter 1. Since input capacitances C_2 of on-path inverter 2 and C_3 of off-path inverter 3 are composed in parallel, the equivalent capacitive load of inverter 1 is their sum $C_2 + C_3$. The capacitors form a **current divider**. If i_1 drives load $C_2 + C_3$, then i_2 must be

$$i_2 = \frac{C_2}{C_2 + C_3} i_1 \, . \tag{6}$$

The larger off-path capacitance C_3 is, the less drive current is available for the path of interest, and the larger is its delay.

In the method of logical effort, we account for the reduced drive current on the path of interest by introducing the **branching effort b** of a logic stage:

$$b = \frac{C_{\text{on-path}} + C_{\text{off-path}}}{C_{\text{on-path}}} \, .$$

Note that $b \geq 1$, and $b = 1$ if $C_{\text{off-path}} = 0$, i.e. when there is no branching. In case of the stage-1 branch in **Figure 3.6**, we have $C_{\text{on-path}} = C_2$ and $C_{\text{off-path}} = C_3$, so that the branching effort at the output of stage 1 of the path is:

$$b_1 = \frac{C_2 + C_3}{C_2} \, .$$

Branching effort b_1 enables us to rewrite Equation (6) as $i_2 = i_1/b_1$. The current dividing branch directs only one b_1^{th} of output current i_1 of inverter 1 to drive inverter 2 on the path of interest. Less drive current on the path of interest increases the effort delay of the stage. Since a stage with branching drives both on-path and off-path load capacitances, the stage effort with a branch is

$$f = gbh\,,$$

rather than $f = gh$, because

$$b \cdot h = \frac{C_{\text{on-path}} + C_{\text{off-path}}}{C_{\text{on-path}}} \cdot \frac{C_{\text{on-path}}}{C_{\text{in}}} = \frac{C_{\text{on-path}} + C_{\text{off-path}}}{C_{\text{in}}}$$

accounts for the actual capacitive load of the stage including off-path load, and assuming that h captures the on-path electrical effort only. C_{in} is the input capacitance of the logic gate that drives the branch.

For n-stage paths, we define the **path branching effort B** as the product of the stage branching efforts:

$$B = \prod_{i=0}^{n} b_i\,,$$

where b_i is the branching effort of stage i. Furthermore, we generalize our definition of **path effort F** to include off-path branching:

$$F = GBH\,,$$

where G is the path logical effort and H the path electrical effort on the path of interest, as before. On a path without branching, we have $B = 1$, and F retains the meaning of our original definition $F = GH$. The minimum path delay of an n-stage path with branching retains its form $\hat{D} = nF^{1/n} + P = n(GBH)^{1/n} + P$, but includes the branching effort in the generalized path effort term.

The concept of branching effort enables us to determine the minimum delay of a path with branching almost as easily as for a path without branching, provided the on-path and off-path capacitances can be scaled in a correlated fashion. Consider 3-stage path $A \rightarrow Y$ in Figure 3.6 again. Without the off-path inverters, the path effort would be $F = GH$, where $G = 1$, since the path consists of inverters with $g_{inv} = 1$ only, and the path electrical effort is $H = C_L/C_1$. Including the off-path inverters requires considering the path branching effort. Stage 1 has branching effort $b_1 = (C_2 + C_3)/C_2$, and the inverter in stage 2 drives on-path load C_4 and off-path load C_5, resulting in branching effort $b_2 = (C_4 + C_5)/C_4$. The path branching effort is the product

$$B = b_1 \cdot b_2 = \frac{(C_2 + C_3)(C_4 + C_5)}{C_2 C_4}\,.$$

With path parasitic delay $P = 3p_{inv} = 3$, the minimum path delay is

$$\hat{D}_{A\to Y} \;=\; 3(GBH)^{1/3} + P \;=\; 3\Big(\frac{(C_2+C_3)(C_4+C_5)C_L}{C_1C_2C_4}\Big)^{\frac{1}{3}} + 3\,.$$

Gate sizing for minimum delay becomes feasible if the on-path and off-path load capacitances of a stage are related. For example, if we are given design constraints $C_3 = C_2$ and $C_5 = 2C_4$, then F reduces to

$$F \;=\; \frac{(C_2+C_2)(C_4+2C_4)C_L}{C_1C_2C_4} \;=\; \frac{2\cdot 3\cdot C_L}{C_1} \;=\; 6H\,.$$

For a given path electrical effort H, we can now apply the method of logical effort to calculate the stage effort $\hat{f} = \sqrt[3]{6H}$ that each stage must bear, and deduce the scale factors to minimize the path delay.

■ **Example 3.10:** Analysis of Branching Circuit

We analyze the delay of path $A \to Y$ of the circuit in **Figure 3.6** and **Experiment 3.4**. Stage-1 inverter 1 is a reference inverter with input capacitance $C_1 = 3$ and scale factor $\gamma_1 = 1$. Assuming the other inverters have variable sizes, the delay of path $A \to Y$ is the sum of the stage delays:

$$\begin{aligned} D_{A\to Y} &= (g_1b_1h_1 + p_1) + (g_2b_2h_2 + p_2) + (g_4b_4h_4 + p_4) \\ &= \left(\frac{C_2+C_3}{C_1} + 1\right) + \left(\frac{C_4+C_5}{C_2} + 1\right) + \left(\frac{C_L}{C_4} + 1\right) \\ &= \gamma_2 + \gamma_3 + \frac{\gamma_4+\gamma_5}{\gamma_2} + \frac{C_L}{3\gamma_4} + 3\,. \end{aligned}$$

You can vary the scale factors $\gamma_2, \ldots, \gamma_5$ and load capacitance C_L in **Experiment 3.4** to verify this path delay expression.

We can determine the minimum delay of path $A \to Y$ for a given load capacitance C_L only, if we also specify the off-path capacitances C_3 and C_5. Fixing these capacitances by specifying constant scale factors for the off-path inverters rarely makes sense, and may even complicate the minimization problem. However, if the design permits scaling both on-path and off-path capacitances of a branch in a correlated fashion, we can apply the method of logical effort directly. Consider the constraints $\gamma_3 = \gamma_2$ and $\gamma_5 = 2\gamma_4$, that we introduced above already. Then, the stage effort that minimizes the path effort is $\hat{f} = \sqrt[3]{6H} = \sqrt[3]{2C_L}$. Given $C_L = 300$, for instance, we find $\hat{f} = 8.43$ and minimum path delay $\hat{D}_{A\to Y} = 3\hat{f} + 3 = 28.3$ delay units. We derive the scale factors γ_2 and γ_4 starting with γ_4. Stage 4 of the path bears stage effort

$$g_4b_4h_4 = 1\cdot 1\cdot \frac{C_L}{C_4}\,,$$

which must be equal to $\hat{f}$, so that

$$\gamma_4 = \frac{C_4}{3} = \frac{C_L}{3\hat{f}} = 11.9\,.$$

Due to constraint $\gamma_5 = 2\gamma_4$, scale factor $\gamma_5 = 23.7$. Analogously, stage 2 bears stage effort

$$g_2 b_2 h_2 = 1 \cdot \frac{C_4 + C_5}{C_4} \cdot \frac{C_4}{C_2}\,.$$

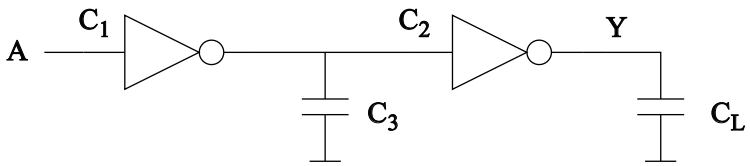
The stage effort must be equal to $\hat{f}$. Rearranging the equation for C_2, we find:

$$\gamma_2 = \frac{C_2}{3} = \frac{C_4}{\hat{f}} = \frac{C_L}{\hat{f}^2} = 4.2\,.$$

According to constraint $\gamma_2 = \gamma_3$, we also have $\gamma_3 = 4.2$. You can verify the minimum path delay in **Experiment 3.4** using rounded scale factors. Furthermore, the summation of the stage delays yields $D_{A\to Y} = 2 \cdot 4.2 + (11.9 + 23.7)/4.2 + 300/(3 \cdot 11.9) + 3 = 28.3$ in agreement with $\hat{D}_{A\to Y}$.

■ **Example 3.11:** Branching Effort Simplies Delay Analysis

Consider the 2-stage path $A \to Y$ below with off-path capacitance C_3. We minimize the delay of the path without using the branching effort first, and then demonstrate how much simpler the analysis becomes when using the branching effort.



Stage 1 of the path consists of an inverter with load capacitance $C_2 + C_3$. Therefore, the stage effort of stage 1 is

$$f_1 = g_1 h_1 = \frac{C_2 + C_3}{C_1}\,,$$

because $g_1 = g_{inv} = 1$. Here, electrical effort h_1 denotes the ratio of the total load capacitance to input capacitance rather than referring to the on-path load capacitance only. The delay of stage 1 is

$$d_1 = f_1 + p_1 = \frac{C_2 + C_3}{C_1} + 1\,,$$

because $p_1 = p_{inv} = 1$. Stage 2 of the path consists of an inverter with load capacitance C_L. Therefore, stage 2 has stage effort

$$f_2 = g_2 h_2 = \frac{C_L}{C_2}$$

and stage delay

$$d_2 = f_2 + p_2 = \frac{C_L}{C_2} + 1\,.$$

The path delay from input A to output Y is the sum of the stage delays

$$D = d_1 + d_2 = \frac{C_2 + C_3}{C_1} + \frac{C_L}{C_2} + 2\,.$$

To minimize D, each stage must bear the same effort. According to the method of logical effort, this occurs under the condition $f_1 + f_2 = 2\sqrt{f_1 \cdot f_2}$. To determine the minimum path effort

$$2\sqrt{f_1 \cdot f_2} = 2\sqrt{\frac{C_2 + C_3}{C_1} \cdot \frac{C_L}{C_2}}$$

we need a constraint that relates on-path capacitance C_2 and off-path capacitance C_3. As a concrete example, assume that $C_3 = 3C_2$. Then, we obtain

$$2\sqrt{f_1 \cdot f_2} = 2\sqrt{\frac{C_2 + 3C_2}{C_1} \cdot \frac{C_L}{C_2}} = 4\sqrt{\frac{C_L}{C_1}}\,.$$

Given load capacitance C_L and input capacitance C_1 of the path, we can determine C_2 by asserting the minimzation condition and rearranging it into a polynomial in C_2:

$$\begin{aligned} f_1 + f_2 &= 2\sqrt{f_1 \cdot f_2} \\ \frac{4C_2}{C_1} + \frac{C_L}{C_2} &= 4\sqrt{\frac{C_L}{C_1}} \\ C_2^2 - \sqrt{C_1 C_L}\, C_2 + \frac{1}{4} C_1 C_L &= 0\,. \end{aligned}$$

The positive root of this quadratic equation is

$$C_2 = \frac{1}{2}\sqrt{C_1 C_L}\,,$$

and the minimum path effort is

$$f_1 + f_2 = 2\sqrt{f_1 \cdot f_2} = 4\sqrt{\frac{C_L}{C_1}}\,.$$

Therefore, the minimum path delay is

$$\hat{D} = f_1 + f_2 + P = 2\sqrt{f_1 \cdot f_2} + P = 4\sqrt{\frac{C_L}{C_1}} + 2\,.$$

Recall that C_L/C_1 is the path electrical effort H of the branching path.

Note that the derivation of $\hat{D}$ requires quite a bit of algebra which increases the chances of introducing errors in the calculation. For comparison, we redo the minimization with the alternative method based on branching efforts. The method of logical effort states that the minimum path delay of a 2-stage path with branching is

$$\hat{D} = 2\sqrt{F} + P\,.$$

In our example, under constraint $C_3 = 3C_2$ the path effort is

$$F = GBH = 1 \cdot \frac{C_2 + C_3}{C_2} \cdot \frac{C_L}{C_1} = 4\frac{C_L}{C_1}\,.$$

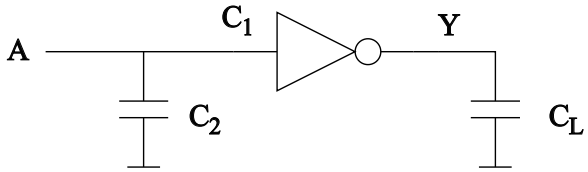
Therefore, the minimum path delay is

$$\hat{D} = 2\sqrt{4\frac{C_L}{C_1}} + 2 = 4\sqrt{\frac{C_L}{C_1}} + 2\,.$$

Note that we arrive at the result by following the method of logical effort like a cookbook recipe, without actually minimizing the delay and laboring through the associated algebra to determine C_2. Using the branching effort simplifies the delay minimization significantly. This is where the method of logical effort begins to shine.

■ **Example 3.12:** Branching Pitfall

When analyzing a branching path with the method of logic effort we account for the branch at the output of a logic stage, because a branch increases the electrical effort and, thus, the delay of the stage. In this example we analyze the branching circuit below, which is the circuit of **Example 3.11** with the stage-1 inverter removed.



Path $A \rightarrow Y$ contains one inverter. Thus, the path has a single stage only. According to the method of logical effort, the minimum delay of the 1-stage path is

$$\hat{D} = F + P = GBH + P = g_{inv} \cdot \frac{C_1 + C_2}{C_1} \cdot \frac{C_L}{C_{in}} + p_{inv}\,,$$

where branching effort $B = (C_1 + C_2)/C_1$ accounts for the branch before the inverter, and C_{in} is the path input capacitance $C_{in} = C_1 + C_2$, because C_1 and C_2 are composed in parallel. Thus, we can simplify $\hat{D}$ to obtain

$$\hat{D} = 1 \cdot \frac{C_1 + C_2}{C_1} \cdot \frac{C_L}{C_1 + C_2} + 1 = \frac{C_L}{C_1} + 1\,.$$

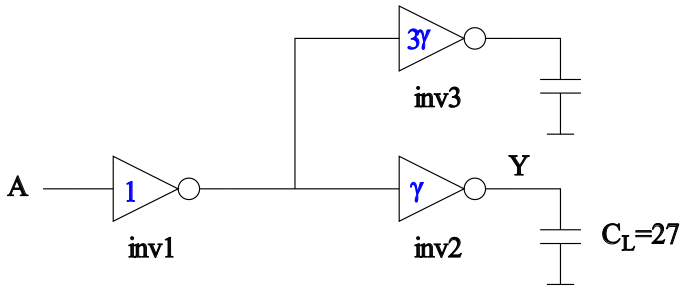
We find that the minimum path delay is independent of off-path capacitance C_2. This result may be not be what we might have expected when comparing to **Example 3.11**, where all off-path capacitances affect the minimum path delay. Thus, we may develop doubts about the correctness of our derivation of $\hat{D}$. Did we do anything wrong in our analysis?

It turns out that the method of logical effort happens to yield the correct result. We just need to interpret $\hat{D}$ carefully. Unlike **Example 3.11** our circuit does not include a logic stage that drives input A. Thus, in our application of the method of logical effort, we included the branching effort perhaps naively. If we interpret the branch as a current divider, then we might conclude that the branch reduces the current that drives the inverter and, hence, slows down the inverter stage. However, this slow

down depends on the driver stage, which the circuit does not specify. Hence, we cannot quantify the slow-down due to the branch as part of the inverter stage after the branch. This argument becomes obvious if we interpret the parallel capacitances of the branch as a current divider, where the important quantity is the voltage of node A rather than the currents into the capacitors. The currents divide such that the voltage across the parallel capacitances remains the same. The voltage of input A changes with a delay proportional to $C_1 + C_2$. Therefore, the branch does not slow down the inverter stage but the speed by which the input voltage of the inverter, and C_2, transitions.

In our analysis with the method of logical effort, the branching effort cancels out input capacitance C_{in}, and the remaining input capacitance C_1 corresponds to the path after the branch. Imagine removing C_2 in the circuit diagram, then the minimum delay of the path is exactly $\hat{D}$ as determined above. The lesson to be learned from this example is that branches do not affect the delay of the path after the branch. Instead, branches effect the delay of their driver stage. If we do not specify the driver stage, we cannot quantify the effect of the branch on the path delay. In the method of logical effort the branch disappears almost magically, just as it should.

Minimize the delay of branching path $A \to Y$ using calculus.



a. For each stage, derive the stage delay.
b. Express path delay D as a function of scale factor γ, and minimize $D(\gamma)$ using ***calculus***.
c. Compute $\hat{D}$ and the associated γ.

Path of interest $A \to Y$ consists of two stages, inverter 1 in stage-1 and inverter 2 in stage-2. Inverter 3 branches off the output of stage 1. Hence, off-path inverter 3 diverts drive current from inverter 1 to inverter 2. We begin with the analysis of stage 1. The load of inverter 1 consists of parallel input capacitances of inverters 2 and 3, that we call C_2 and C_3. Therefore, the electrical effort of inverter 1 with input capacitance C_1, counting both on-path and off-path loads, is

$$h_1 = \frac{C_2 + C_3}{C_1},$$

and the delay of stage 1 is

$$d_1 = g_{inv} h_1 + p_{inv} = \frac{C_2 + C_3}{C_1} + 1.$$

Since the scale factors of the inverters are given in the schematic, we know that inverter 1 is a reference inverter with $C_1 = C_{inv} = 3$ capacitive units, inverter 2 has input capacitance $C_2 = \gamma C_{inv} = 3\gamma$, and inverter 3 has input capacitance $C_3 = 9\gamma$. Substituting the capacitances in d_1, we obtain

$$d_1 = \frac{3\gamma + 9\gamma}{3} + 1 = 4\gamma + 1\,.$$

Since stage 2 does not branch, the delay of inverter 2 is

$$d_2 = g_{inv}\frac{C_L}{C_2} + p_{inv} = \frac{27}{3\gamma} + 1 = \frac{9}{\gamma} + 1\,.$$

The delay of path $A \to Y$ is the sum of the stage delays:

$$D = d_1 + d_2 = 4\gamma + \frac{9}{\gamma} + 2\,.$$

The path delay is a function of scale factor γ. Therefore, we can minimize the path delay by setting the derivative to zero:

$$\frac{dD}{d\gamma} = 4 - \frac{9}{\gamma^2} \stackrel{!}{=} 0$$
$$\gamma = \frac{3}{2}\,.$$

Thus, the minimum path delay from A to Y is

$$\hat{D} = D(3/2) = 14$$

time units.

Hide

Minimize the delay of branching path $A \to Y$ in **Exercise 3.8** using the method of logical effort.

a. Determine path logical effort G, path electrical effort H, path branching effort B, and path parasitic delay P.
b. Compute best stage effort $\hat{f} = \sqrt{GBH}$.
c. Compute minimum path delay $\hat{D} = 2\hat{f} + P$ and associated scale factor γ.

We begin by determining the branching efforts of the stages on the path of interest. Inverter 1 in stage 1 drives on-path inverter 2 and off-path inverter 3. The input capacitances of these inverters are $C_1 = C_{inv} = 3$, $C_2 = \gamma C_{inv} = 3\gamma$, and $C_3 = 9\gamma$. Branching effort b_1 of stage 1 is the ratio of the on-path and off-path load capacitances:

$$\begin{aligned} b_1 &= \frac{C_{on-path} + C_{off-path}}{C_{on-path}} \\ &= \frac{C_2 + C_3}{C_2} \\ &= \frac{3\gamma + 9\gamma}{3\gamma} \\ &= 4\,. \end{aligned}$$

Stage 2 on the path of interest consists of inverter 2. It's output is connected to load capacitance C_L without branching. Therefore, we account for branching effort b_2 of stage 2 as

$$b_2 = 1\,.$$

The path branching effort of path $A \to Y$ is the product of the stage branching efforts:

$$B = b_1 b_2 = 4 \cdot 1 = 4\,.$$

The path logical effort of path $A \to Y$ is the product of the stage logical efforts:

$$G = g_1 g_2 = g_{inv} g_{inv} = 1\,.$$

The path electrical effort of path $A \to Y$ is the product of the stage electrical efforts:

$$H = h_1 h_2 = \frac{C_2}{C_1}\frac{C_L}{C_2} = \frac{C_L}{C_1} = \frac{27}{3} = 9\,.$$

Note that we use the on-path electrical efforts for the stage efforts h_1 and h_2 as if there were no off-path branches. Branches are accounted for separately with branching effort B.

The path parasitic delay of path $A \to Y$ is the sum of the stage parasitic delays:

$$P = p_1 + p_2 = p_{inv} + p_{inv} = 2\,.$$

Path effort F of a path with branching is $F = GBH$, and best stage effort $\hat{f} = \sqrt[n]{F}$. For our 2-stage path $A \to Y$, we find

$$\hat{f} = \sqrt{GBH} = \sqrt{1 \cdot 4 \cdot 9} = 6\,.$$

We obtain minimum path delay $\hat{D}$ if each stage of the path bears best stage effort $\hat{f}$. Then, we have for our 2-stage path:

$$\hat{D} = 2\hat{f} + P = 2 \cdot 6 + 2 = 14\,.$$

This is the same result that we found through the minimization by calculus in **Exercise 3.8**, however, with straightforward algebra due to the method of logical effort.

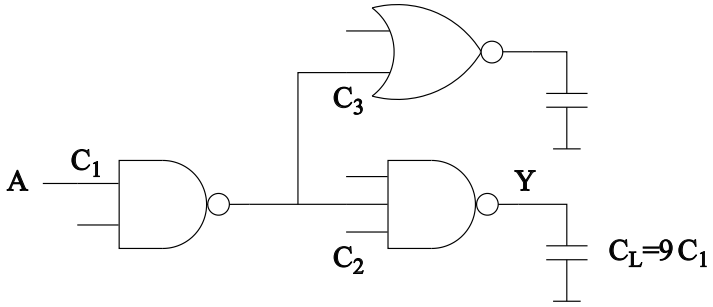
To determine scale factor γ, we notice that each stage on path $A \to Y$ must bear best stage effort $\hat{f} = 6$. In particular, the stage effort of inverter 2 must be equal to $\hat{f}$, which yields an equation for γ:

$$f_2 = g_2 b_2 h_2 = g_{inv} b_2 \frac{C_L}{C_2} \stackrel{!}{=} \hat{f}$$
$$1 \cdot 1 \cdot \frac{27}{3\gamma} = 6$$
$$\gamma = 3/2\,.$$

Alternatively, we could use the stage effort of inverter 1, $g_1 b_1 h_1 \stackrel{!}{=} \hat{f}$, to deduce $\gamma = 3/2$.

Hide

Use the method of logical effort to minimize the delay of branching path $A \to Y$ assuming $C_3 = 4C_2$.



a. Determine path logical effort G, path electrical effort H, path branching effort B, and path parasitic delay P.
b. Compute best stage effort $\hat{f}$.
c. Compute minimum path delay $\hat{D}$.

Path of interest $A \to Y$ consists of two stages, i.e. $n = 2$. Stage 1 consists of a 2-input NAND gate with input capacitance C_1, and stage 2 features a 3-input NAND gate with input capacitance C_2. The 2-input NOR gate with input capacitance C_3 branches of the output of stage 1. We analyze the circuit, starting with the path logical effort of path $A \to Y$:

$$G = g_1 g_2 = g_{nand2} g_{nand3} = \frac{4}{3}\frac{5}{3} = \frac{20}{9}\,.$$

The path electrical effort of path $A \to Y$ is the ratio of load capacitance and path input capacitance:

$$H = \frac{C_L}{C_1} = \frac{9C_1}{C_1} = 9\,.$$

Path branching effort B is the product of the stage branching efforts. Given that $C_3 = 4C_2$, we find:

$$B = b_1 b_2 = \frac{C_2 + C_3}{C_2} \cdot 1 = \frac{C_2 + 4C_2}{C_2} = 5\,.$$

The path parasitic delay is the sum of the stage parasitic delays:

$$P = p_1 + p_2 = p_{nand2} + p_{nand3} = 2 + 3 = 5\,.$$

Since the path has two stages, the best stage effort is

$$\hat{f} = \sqrt{GBH} = \sqrt{\frac{20}{9} \cdot 5 \cdot 9} = 10\,,$$

and the minimum path delay

$$\hat{D} = 2\hat{f} + P = 2 \cdot 10 + 5 = 25\,.$$

We note that path $A \to Y$ is overburdened, because $\hat{f} = 10 > f^* = 3.59$. Thus, we should be able to reduce the path delay below $\hat{D}$ by path sizing.

Hide

3.5. Forks

Branching circuits occur in countless applications, for example to drive the complemented and uncomplemented inputs of an ***XOR gate***. If we have a circuit that produces signal A, we need to generate $\overline{A}$ to drive both the A and $\overline{A}$ inputs of the XOR gate. A branching circuit is a **fork**, if it branches to output both the complemented and uncomplemented input. More specifically, we define an **n-fork** for $n \geq 1$ as a fork with two legs, one with n inverters and the other with $n-1$ inverters.

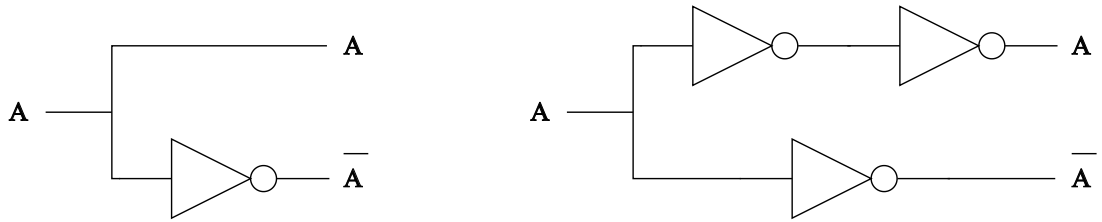


Figure 3.7: A 1-fork (left) and a 2-fork (right) output both the complemented and uncomplemented input.

Depending on the choice of n, one of the legs, the *even leg*, features an even number of inverters while the *odd leg* has an odd number of inverters. Since the number of stages in the even and odd legs differ, the question is how to minimize the delay of the n-fork. At the first glance, we may identify the longer n-stage leg as the path of interest, and minimize its path delay. Since the shorter leg branches off the longer leg, we are in need of a constraint to correlate the sizes of the stage-1 inverters of the legs. In terms of branching effort, the stage-1 inverter of the leg with n inverters presents the on-path capacitance, and the stage-1 inverter of the leg with $n-1$ stages the off-path capacitance. In the following, we derive a perhaps less obvious constraint for the on-path and off-path capacitances that minimizes the delay of an n-fork by equalizing the delays of both legs.

3.5.1. Fork Delay

We study n-forks with an n-stage leg and an $(n-1)$-stage leg as shown in **Figure 3.8**, strictly speaking for $n \geq 2$. We assume that the on-path electrical efforts for the legs are given as H_1 for leg 1 and H_2 for leg 2. We express the load capacitances of the legs as a function of fork input capacitance C_{in}. Leg 1 is the n-stage leg. The stage-1 inverter of leg 1 has input capacitance C_1 and load capacitance $C_{L,1} = H_1 C_{in}$. Leg 2 is the $n-1$-stage leg. Its stage-1 inverter has input capacitance C_2, and the load capacitance is $C_{L,2} = H_2 C_{in}$. From the perspective of the fork input, the legs present a parallel composition of input capacitances, so that

$$C_{in} = C_1 + C_2 \,.$$

Since a fork branches into two legs, the path effort of leg 1 is

$$F_1 = G_1 B_1 H_1 = 1 \cdot \frac{C_1 + C_2}{C_1} \cdot \frac{C_{L,1}}{C_{in}} = H_1 \frac{C_{in}}{C_1} \,,$$

and of leg 2

$$F_2 = G_2 B_2 H_2 = 1 \cdot \frac{C_1 + C_2}{C_2} \cdot \frac{C_{L,2}}{C_{in}} = H_2 \frac{C_{in}}{C_2} \,.$$

If we replace the fork with an equivalent circuit, the equivalent circuit has input capacitance C_{in} and output capacitance $C_L = C_{L,1} + C_{L,2} = (H_1 + H_2) C_{in}$. Thus, the electrical effort of the entire fork or its equivalent circuit is $H = C_L / C_{in} = H_1 + H_2$.

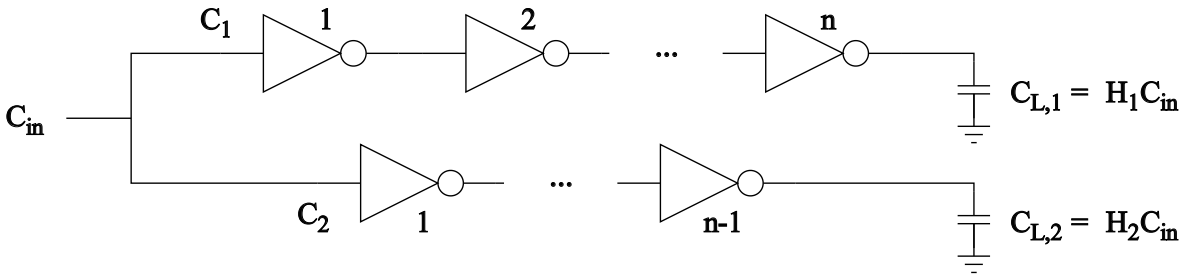


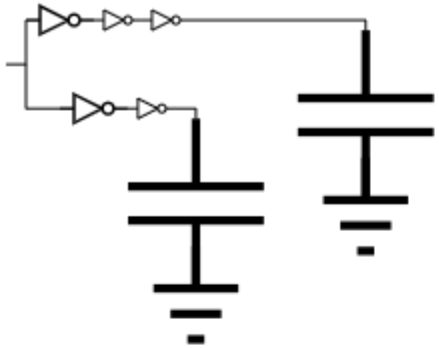
Figure 3.8: An n-fork with input capacitance $C_{in} = C_1 + C_2$ and load capacitance $C_L = (H_1 + H_2) C_{in}$.

To keep things simple, we begin our study of forks by assuming that the capacitive loads of both legs are equal, i.e. $C_{L,1} = C_{L,2} = H_l C_{in}$, where leg electrical effort $H_l = H_1 = H_2$. Thus, the total load of the fork is $C_L = 2 H_l C_{in}$. The path efforts of the legs, $F_1 = H_l C_{in} / C_1$ and $F_2 = H_l C_{in} / C_2$, may differ, depending on the choice of leg input capacitances C_1 and C_2. In fact, since the lengths of the legs differ by one stage, the longer leg 1 can support a larger path effort than the shorter leg 2. Given equal load capacitances, we may increase the path effort of leg 1 by reducing C_1. If we require a constant input capacitance C_{in}, decreasing C_1 must be

balanced by increasing C_2. A shift of input capacitance between the two legs corresponds to a proportional shift in the division of the drive current between the legs. **Experiment 3.5** allows you to vary the input capacitances C_1 and C_2 of the legs, and observe the effect on the fork delay.

▶ **Experiment 3.5:** Delay of a 3-Fork with Equal Loads

Assume that input capacitance C_{in} is fixed, for example by other circuit design constraints. Then, the leg input capacitances C_1 and C_2 are correlated through the parallel composition: $C_1 + C_2 = C_{in}$. Choose C_{in} and total load $C_L = 2H_lC_{in}$ by tuning leg effort H_l. Minimize the fork delay by sizing the inverters.



Inverter 1: 2 Inverter 3: 1 Inverter 5: 1

Inverter 2: 2 Inverter 4: 1

Leg effort: 25 Total Load: 600 Delay: 104.5

The key observation for minimizing the fork delay in **Experiment 3.5** is that the allocation of input capacitance C_{in} to leg input capacitances C_1 and C_2 matters. Of course, this fact applies to any branching circuit, not only to forks. In case of a fork with equal loads, the correlation between the input capacitances when keeping $C_{in} = C_1 + C_2$ constant enables us gain additional insight into the the cause of the fork delay. The delay of an n-stage inverter chain is minimal, if each stage bears the same effort $\hat{f} = \sqrt[n]{H}$. Each leg of a fork is an inverter chain. If we size the inverters to minimize the leg delays, we obtain minimum delay $\hat{D}_1(n) = n\sqrt[n]{F_1} + n$ for leg 1, and minimum delay $\hat{D}_2(n-1) = (n-1)\sqrt[n-1]{F_2} + (n-1)$ for leg 2. We may be satisfied with a delay of the fork that is the maximum of the independently minimized leg delays, $\max(\hat{D}_1, \hat{D}_2)$. However, we can do better by recognizing that the minimum fork delay occurs if the leg delays are equal. If they are not, we can reduce the larger leg delay at the expense of increasing the smaller delay of the other leg.

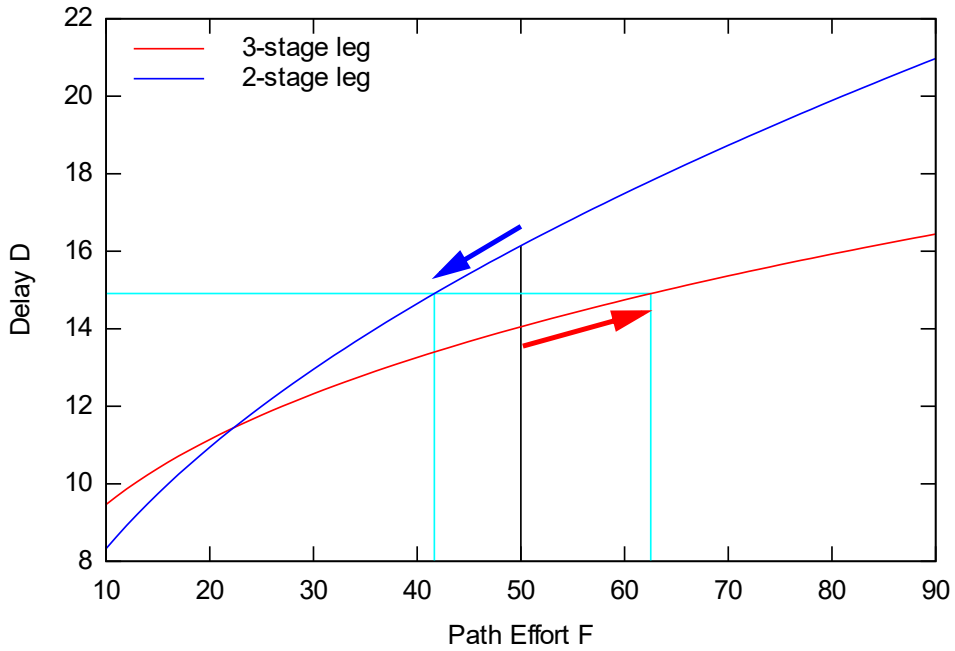


Figure 3.9: Minimize the delay of a 3-fork by shifting path efforts between the legs.

Figure 3.9 illustrates the idea for a 3-fork with leg electrical effort $H_l = H_1 = H_2 = 25$ and input capacitance $C_{in} = 12$. In this case, the load capacitances are $C_{L,1} = C_{L,2} = 300$. First, assume that the legs present equal loads of $C_1 = C_2 = C_{in}/2 = 6$ capacitive units each. Then, both legs bear equal path efforts $F_1 = F_2 = 50$. The plot shows the minimum path delays $\hat{D}_1(3)$ of the 3-stage leg and $\hat{D}_2(2)$ of the 2-stage leg as a function of path effort F. For $F = 50$, 3-stage leg 1 has minimum delay $\hat{D}_1(3) = 14.05$, and is faster than 2-stage leg 2 with minimum delay $\hat{D}_2(2) = 16.14$. The fork delay is dominated by leg 2. We can minimize the fork delay by reducing path effort F_2 of leg 2 at the expense of increasing path effort F_1 of leg 1 to honor the constraint $C_{in} = C_1 + C_2$. When reducing $F_2 = C_{L,2}/C_2$ by increasing input capacitance C_2 of leg 2, we reduce C_1 by the same amount. In turn, the reduction of C_1 increases $F_1 = C_{L,1}/C_1$ and,

consequently, leg delay $\hat{D}_1(3)$. The fork delay is minimal when the leg delays are equal, $\hat{D}_1(3) = \hat{D}_2(2)$. In **Figure 3.9** this occurs when $F_1 = 62.5$ and $F_2 = 41.6$. The corresponding input capacitances are $C_1 = 4.8$ for leg 1 and $C_2 = 7.2$ for leg 2. As a result the minimum fork delay amounts to 14.9 delay units.

We can determine the leg input capacitances for minimum fork delay by formalizing the idea of shifting input capacitance between the legs. To that end, we introduce **skew factor α** in range $0 < \alpha < 1$, and express the leg input capacitances as fractions of the fork input capacitance:

$$C_1 = \alpha C_{in}\,, \qquad C_2 = (1-\alpha) C_{in}\,,$$

such that $C_1 + C_2 = C_{in}$. Now, the path efforts are correlated through skew factor α. For leg 1 we have $F_1 = H_1/\alpha$, and for leg 2 $F_2 = H_2/(1-\alpha)$. The key insight for minimizing the fork delay is to **equalize the leg delays** rather than minimizing the leg delays independently:

$$\hat{D}_1(n) = n\left(\frac{H_1}{\alpha}\right)^{\frac{1}{n}} + n \quad \stackrel{!}{=} \quad (n-1)\left(\frac{H_2}{1-\alpha}\right)^{\frac{1}{n-1}} + (n-1) = \hat{D}_2(n-1)\,.$$

Given leg electrical efforts H_1 and H_2 for an n-fork, even in the special case with equal loads, $H_1 = H_2 = H_l$, this constraint presents a nonlinear equation in α. Solving this equation for α analytically is difficult for arbitrary n. We could determine α numerically, for example by means of a Newton iteration. Alternatively, we can reduce the problem of computing α to finding the ***roots of a polynomial***, for which solutions exist as part of mathematical software packages like **Octave** or **Mathematica** with online portal **WolframAlpha**.

3.5.2. 1d-Analysis Method

The method of logical effort does not require explicit use of skew factor α to derive the minimum fork delay. We demonstrate this derivation by means of the 3-fork in **Figure 3.10** below. First, we apply the key insight of the method of logical effort, i.e. a path achieves minimum delay if all stages bear the same effort. This insight enables us to allocate stage efforts to each inverter as follows. Consider the 3-stage leg of the 3-fork, and think of the stage effort as effort delay in units of time. If the effort delay of the whole leg is d_1, then each of the 3 stages must have effort delay $d_1/3$ to minimize the path delay. Analogously, if the 2-stage leg has effort delay d_2, then each stage must have effort delay $d_2/2$ to minimize the path delay. Then, the leg delays are

$$\begin{aligned}\hat{D}_1(3) &= 3\frac{d_1}{3} + 3 \\ \hat{D}_2(2) &= 2\frac{d_2}{2} + 2\,.\end{aligned}$$

Second, we apply our key insight about minimizing fork delay, i.e. both legs must have equal path delay. In case of the 3-fork, we demand

$$3\frac{d_1}{3}+3=2\frac{d_2}{2}+2\,.$$

This equality is satisfied by an effort delay d, if $d_1=d$ and $d_2=d+1$, because

$$\begin{aligned} 3\frac{d}{3}+3 &= 2\frac{d+1}{2}+2 \\ \Leftrightarrow \quad d+3 &= d+1+2\,. \end{aligned}$$

Figure 3.10 annotates the inverters with the effort delays that yield minimum fork delay as a function of delay d.

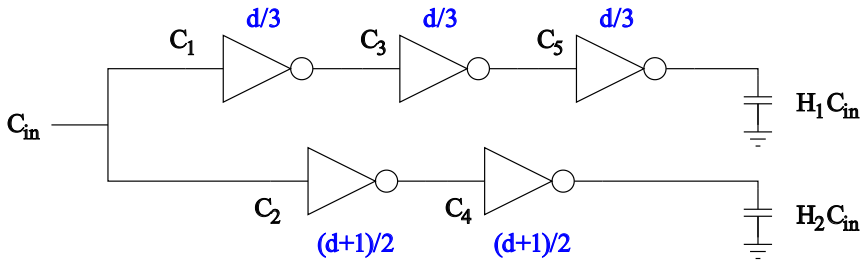


Figure 3.10: 3-fork with effort delay allocation.

Two aspects of the **effort delay allocation** in **Figure 3.10** deserve a closer look. First, the effort delays minimize the fork delay by equalizing the leg delays, which determines skew factor α implicitly, and by minimizing the path delay of each leg, because each gate on a leg bears the same stage effort. To solve the delay minimization problem, we merely need to determine the value of d. Second, our algebraic parameterization of leg effort delays $d_1(d)=d$ and $d_2(d)=d+1$ may be viewed as compensating for one unit of parasitic delay on leg 1 with one unit of effort delay on leg 2. The path parasitic delay of leg 1 is $P_1=3p_{inv}=3$, whereas leg 2 has path parasitic delay $P_2=2p_{inv}=2$. We call the difference $\Delta p=P_1-P_2=1$ in $d_2(d)=d+\Delta p$ the **parasitic delay compensation**, because it compensates the excess parasitic delay of leg 1 with effort delay in leg 2.

Next, we analyze the delay of the fork legs as a function of d. The effort delay of an inverter, $f=g_{inv}h$, is equal to the electrical effort, because the logical effort is $g_{inv}=1$. Thus, for the inverters of leg 1, we find for the stage-1 inverter effort delay

$$\frac{d}{3}=\frac{C_3}{C_1}\,,$$

for the stage-2 inverter effort delay

$$\frac{d}{3}=\frac{C_5}{C_3}\,,$$

and for the stage-3 inverter effort delay

$$\frac{d}{3} = \frac{H_1 C_{in}}{C_5}\,.$$

Note that the product of the right-hand sides is a telescoping product. Therefore, multiplying the equations yields

$$\frac{d^3}{27} = \frac{H_1 C_{in}}{C_1}\,. \tag{7}$$

Analogously, for the inverters of leg 2, we find the effort delay of the stage-1 inverter

$$\frac{d+1}{2} = \frac{C_4}{C_2}\,,$$

the effort delay of the stage-2 inverter

$$\frac{d+1}{2} = \frac{H_2 C_{in}}{C_4}$$

and the product of the equations

$$\frac{(d+1)^2}{4} = \frac{H_2 C_{in}}{C_2}\,. \tag{8}$$

The input capacitance of the fork is the sum of the leg input capacitances. Substituting the expressions in Equation **(7)** and Equation **(8)** for C_1 and C_2, we find

$$C_{in} = C_1 + C_2 = \frac{27 H_1 C_{in}}{d^3} + \frac{4 H_1 C_{in}}{(d+1)^2}\,.$$

Rearranging the equality we obtain a polynomial in variable d:

$$d^5 + 2d^4 + (1 - 4H_2)d^3 - 27H_1 d^2 - 54H_1 d - 27H_1 = 0\,.$$

The solutions to this equation are the ***roots of the polynomial***, i.e. those values of d where the polynomial is zero. A polynomial of degree 5 has 5 roots. Given H_1 and H_2, we can determine the roots with a mathematical software package like **WolframAlpha**. For $H_1 = H_2 = 25$, copy or type this equation:

```
d^5 + 2 d^4 + (1 - 4 * 25) d^3 - 27 * 25 d^2 - 54 * 25 d - 27 * 25 = 0
```

into the **WolframAlpha** input form and click the equal sign.

WolframAlpha returns the roots:

$$\begin{aligned} d_1 &= -2.53588 \\ d_2 &= -0.750052 \\ d_3 &= 11.9076 \\ d_4 &= -5.31085 - 1.264i \\ d_5 &= -5.31085 + 1.264i\,. \end{aligned}$$

Root d_3 is the only feasible solution for effort delay d, which must be a positive real number to qualify as a time. Knowing that $d = 11.91$, we can deduce the design parameters of our fork. The 3-fork has a minimum delay of $\hat{D}_1(3) = \hat{D}_2(2) = d + 3 = 14.91$ delay units, if we scale the stage-1 inverters of the legs to present input capacitances $C_1 = 4.8$ according to Equation **(7)**, $C_2 = 7.2$ according to Equation **(8)**, and all other inverters for the input capacitances derived above. Note that skew factor $\alpha = C_1/C_{in} = 0.4$, although we do not need to know α to minimize the fork delay. **Experiment 3.5** enables you to verify our fork design.

The methodology for deriving the parameters of a fork with minimum delay generalizes to branching circuits with two legs. We call it the *1d-analysis method*, because it is based on finding the roots of a univariate polynomial with effort delay d as the only variable.

1d-Analysis Method

Given a branching circuit with an n-stage leg 1 and an m-stage leg 2, input capacitance $C_{in} = C_1 + C_2$, and load capacitances $H_1 C_{in}$ and $H_2 C_{in}$. Follow these steps to minimize the circuit delay:

1. Determine the leg parasitic delays P_1 and P_2.
2. Without loss of generality, assume that $P_1 \geq P_2$, and *parasitic delay compensation* $\Delta p = P_1 - P_2$. Allocate effort delay d/n to each stage of leg 1 and $(d + \Delta p)/m$ to each stage of leg 2.
3. Form the telescoping products of the leg effort delays, including logical efforts:

$$\left(\frac{d}{n}\right)^n = \frac{G_1 H_1 C_{in}}{C_1}\,, \qquad \left(\frac{d + \Delta p}{m}\right)^m = \frac{G_2 H_2 C_{in}}{C_2}\,.$$

Substitute leg input capacitances C_1 and C_2 in the branch constraint $C_{in} = C_1 + C_2$ to derive a univariate polynomial and determine its roots:

$$d^n(d+\Delta p)^m - m^m G_2 H_2 d^n - (d+\Delta p)^m n^n G_1 H_1 = 0\,.$$

4. Identify the positive real root d that solves our problem, and determine the scale factors of all gates. The minimum circuit delay is $d + P_1 = (d + \Delta p) + P_2$.

The 1d-analysis method enables us to analyze the 1-fork with a 0-stage leg and equal loads in **Figure 3.11**. The unassuming circuit designer might be tempted to use this degraded fork to generate the complemented and uncomplemented input. However, a 1-fork is rarely a good circuit to use.

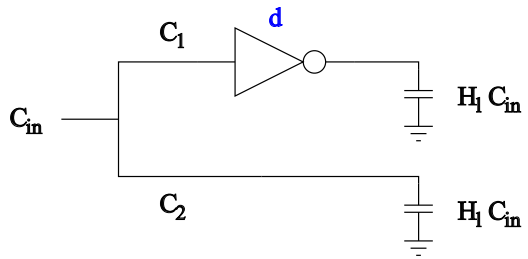


Figure 3.11: A 1-fork cannot equalize leg delays.

According to the 1d-analysis method, we assign effort delay d to the inverter in leg 1. Then, leg 1 has delay $D_1 = d + 1$ and leg 2 has delay $D_2 = 0$. Note that negative effort delay $d = -1$ would be required to equalize the leg delays. Furthermore, the effort delay of leg 1 is $d = H_l C_{in}/C_1$, and the input capacitance of leg 2 is the load capacitance, $C_2 = H_l C_{in}$. Thus, the branch constraint yields

$$\begin{aligned} C_{in} &= C_1 + C_2 = \frac{H_l C_{in}}{d} + H_l C_{in} \\ \Leftrightarrow \quad d &= \frac{H_l}{1 - H_l}\,. \end{aligned}$$

Since CMOS inverters have a positive effort delay, $d > 0$, we conclude that $1 - H_l > 0$, and obtain the constraint on the leg electrical effort:

$$0 < H_l < 1\,.$$

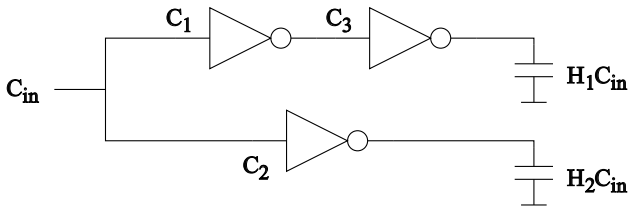
This constraint determines the input capacitance of the inverter in leg 1. For example, leg electrical effort $H_l = 3/4$ yields

$$C_1 = C_{in} - C_2 = C_{in} - \frac{3}{4}C_{in} = \frac{1}{4}C_{in}\,.$$

In terms of C_1, we can express leg load $H_l C_{in} = (3/4)C_{in} = 3C_1$ and input capacitance $C_{in} = 4C_1$. Thus, all capacitances of the 1-fork are determined by choosing H_l. As a consequence, effort delay d of the inverter in leg 1 is $d = 3C_1/C_1 = 3$, and the delay of leg 1 is $D_1 = d + p_{inv} = 4$ time units. This style of 1d-

analysis shows that the 1-fork is effective for $H_l \approx 1/2$. However, if H_l approaches value 1, the inverter has to stem an infinitely large effort. The underlying problem is that the 1-fork has only one inverter to equalize the leg delays. Therefore, as a rule of thumb, it is rarely a good idea to use a 1-fork. Use a 2-fork instead.

Use the ***1d-analysis method*** to minimize the delay of the 2-fork assuming $H_1 = 10$ and $H_2 = 15$.



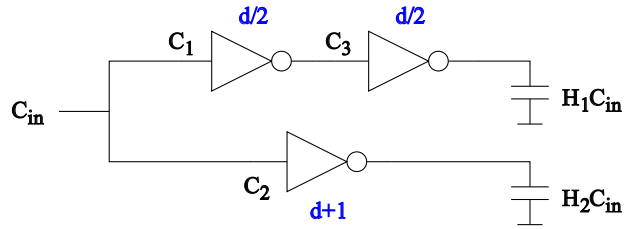
a. Determine the leg parasitic delays and allocate effort delays using one parameter d.
b. For each leg, form the products of the effort delays, and derive a polynomial in d.
c. Determine the root of the polynomial and the minimum leg delays $\hat{D}_1$ and $\hat{D}_2$.

The 2-fork has two inverters on leg 1 and one inverter on leg 2. The parasitic delay of leg 1 is $P_1 = 2p_{inv} = 2$ and of leg 2 $P_2 = p_{inv} = 1$. The difference

$$\Delta p = P_1 - P_2 = 1$$

s the parasitic delay compensation that we allocate in the stage effort assignments to balance the leg delays.

Assuming that leg 1 has a total effort delay of d, we allocate stage effort delay $d/2$ to each of the two inverters, because path delay is minimized if all stages bear the same effort. Since leg 2 has a single stage only, the inverter has to bear all of the path effort plus the parasitic delay compensation, which amounts to stage effort $d + 1$ for the inverter in leg 2.



Given the allocation of effort delays, we form the products of the stage effort delays, which are simple expressions because the product of the stage electrical efforts telescopes. For eg 1, we obtain

$$\left(\frac{d}{2}\right)^2 = \frac{H_1 C_{in}}{C_1}.$$

Leg 2 has one stage only, and the stage effort equals the electrical effort because $g_{inv} = 1$:

$$d+1 = \frac{H_2 C_{in}}{C_2}\,.$$

We rearrange these equations to obtain expressions for C_1 and C_2:

$$C_1 = \frac{4H_1}{d^2} C_{in}\,, \qquad C_2 = \frac{H_2}{d+1} C_{in}\,.$$

Next, we substitute C_1 and C_2 in the branch constraint, $C_{in} = C_1 + C_2$, and rearrange the equation into a polynomial:

$$\begin{aligned} C_{in} &= C_1 + C_2 \\ C_{in} &= \frac{4H_1}{d^2} C_{in} + \frac{H_2}{d+1} C_{in} \\ d^2(d+1) &= 4H_1(d+1) + H_2 d^2 \\ d^3 + (1-H_2)d^2 - 4H_1 d - 4H_1 &= 0\,. \end{aligned}$$

Effort delay d is a root of the polynomial

$$d^3 - 14d^2 - 40d - 40\,,$$

given $H_1 = 10$ and $H_2 = 15$. This polynomial has two complex roots and one real root:

$$d = 16.56\,.$$

Therefore, leg 1 has a delay of $\hat{D}_1 = d + P_1 = 16.56 + 2 = 18.56$ time units and leg 2 has the same delay $\hat{D}_2 = (d+1) + P_2 = 17.56 + 1 = 18.56$ time units.

Hide

3.6. Branching Circuits

The **1d-analysis method** enables us to design not only forks but also more general instances of branching circuits with minimum delay. In this section, we discuss design issues of branches with logic gates, how to determine the best number of stages of branches, how to handle more than two branches, and branches that reconverge again.

3.6.1. Path Sizing Branches

In Section ***Forks***, we minimize the delay of a 3-fork. We did not ask whether the 3-fork has appropriate leg lengths, or whether a 2-fork or 4-fork would result in an even smaller delay for the given load. In the following, we examine the path sizing problem for a branching circuit if both legs are inverter chains of equal length, see **Figure 3.12**.

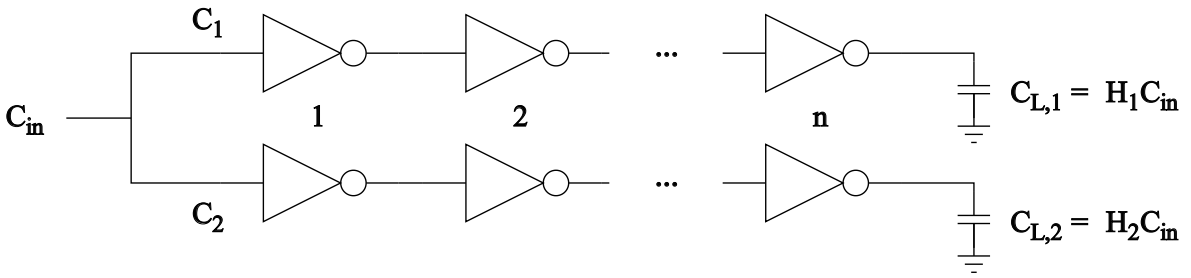


Figure 3.12: A branching circuit with two n-stage legs, input capacitance $C_{in} = C_1 + C_2$ and load capacitance $C_L = (H_1 + H_2)C_{in}$.

If the electrical efforts of the legs were equal, $H_l = H_1 = H_2$, the circuit is symmetric. We would choose equal leg input capacitances, $C_1 = C_2$, to supply both legs with equal drive currents. According to the branch constraint, $C_{in} = C_1 + C_2$, and given input capacitance C_{in}, we scale the stage-1 inverters of the legs such that $C_1 = C_2 = C_{in}/2$. Since both legs bear the same electrical effort, we can determine the best number of stages n using the **path sizing calculator** with path effort $F = 2H_l$.

In general, the leg electrical efforts may differ such that $H_1 \neq H_2$. According to the method of logical effort, we minimize the delay of each leg, independently, if each inverter stage of a leg bears the same effort. For leg 1, given path effort $F_1 = G_1 B_1 H_1 = C_{L,1}/C_1$, the stage effort that minimizes the leg delay is $\hat{f}_1 = \sqrt[n]{C_{L,1}/C_1}$. Analogously, the stage effort for leg 2 is $\hat{f}_2 = \sqrt[n]{C_{L,2}/C_2}$. To minimize the delay of the entire branching circuit we equalize the leg delays, a lesson we have learned from the ***design of forks***.

$$\hat{D}_1(n) = n\left(\frac{C_{L,1}}{C_1}\right)^{\frac{1}{n}} + n \quad \stackrel{!}{=} \quad n\left(\frac{C_{L,2}}{C_2}\right)^{\frac{1}{n}} + n = \hat{D}_2(n)\,.$$

Since both legs have the same number of stages n, this constraint simplifies to

$$\frac{C_1}{C_2} = \frac{H_1}{H_2}\,. \tag{9}$$

We find that we can minimize the branch delay by choosing the leg input capacitances proportional to the leg electrical efforts. Given the branch constraint with input capacitance $C_{in} = C_1 + C_2$, we obtain the leg input capacitances:

$$C_1 = \frac{H_1}{H_1 + H_2} C_{in}\,, \qquad C_2 = \frac{H_2}{H_1 + H_2} C_{in}\,.$$

Furthermore, Equation **(9)** yields

$$\hat{f}_1 = \hat{f}_2 = \sqrt[n]{H_1 + H_2}\,,$$

i.e. we minimize the delay of the branching circuit, if the stage efforts in the two legs are equal.

■ **Example 3.13:** Path Sizing Branches of Inverter Chains

We solve the path sizing problem of the branching circuit in **Figure 3.12** for input capacitance $C_{in} = 12$ and load capacitances $C_{L,1} = 600$ and $C_{L,2} = 300$.

First, we deduce the leg electrical efforts $H_1 = C_{L,1}/C_{in} = 50$ and $H_2 = C_{L,2}/C_{in} = 25$. Second, due to Equation **(9)**, we notice that input capacitance C_1 of leg 1 should be twice as large as C_2. Furthermore, since we are given $C_{in} = 12$, we can calculate in leg input capacitances:

$$C_1 = \frac{50}{50+25} \cdot 12 = 8\,, \qquad C_2 = \frac{25}{50+25} \cdot 12 = 4\,.$$

Third, the path efforts of the legs are equal, and amount to $F_1 = F_2 = H_1 + H_2 = 75$. With the aid of the **path sizing calculator**, we find that we should use $n = 3$ stages in each leg. Then, we minimize the delay of the circuit by sizing the six inverters such that each inverter bears stage effort $\hat{f} = \sqrt[3]{75} = 4.2$. The minimum delay is $\hat{D} = 3\hat{f} + 3 = 15.65$ delay units.

Let us contemplate the solution to the path sizing problem for the branching circuit in **Example 3.13**: We scale the input capacitances of the stage-1 inverters in proportion to the path electrical efforts. As a result, we equalize the path efforts of the legs, which enables us to determine the path lengths of both legs simultaneously, as if we were sizing a single path. This observation motivates us to introduce the concept of an **equivalent inverter path**, in analogy to equivalent ***resistive*** and ***capacitive*** networks. **Figure 3.13** shows the equivalent inverter path of the branching circuit in **Figure 3.12** with input capacitance C_{in} and the sum of the leg electrical efforts $H_1 + H_2$ as path electrical effort.

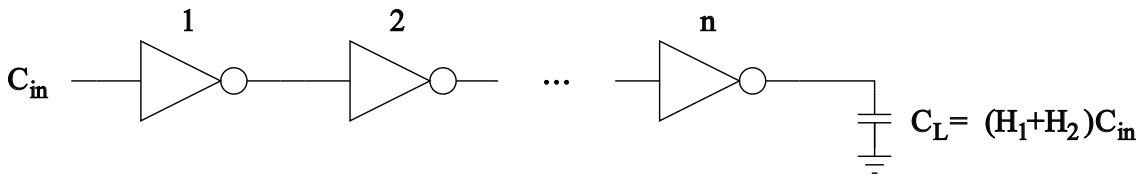


Figure 3.13: Equivalent inverter path of the branching circuit in **Figure 3.12**.

The equivalent inverter path enables us to reduce the path sizing problem for a branching circuit to that of a multistage path as discussed in Section ***Path Sizing***. Once we have determined the best number of stages, we proceed by employing Equation **(9)** to scale the stage-1 inverters of the legs, and perform gate sizing

according to the method of logical effort to minimize the path delay of the branching circuit. This method generalizes to branching circuits with arbitrary logic gates, if we replace the leg electrical efforts with the leg path efforts to account for the leg logical efforts.

For branching circuits with logic gates on the legs, the path efforts of the legs are:

$$\begin{aligned} F_1 &= G_1 B_1 H_1 = G_1 \frac{C_{L,1}}{C_{in}} = G_1 H_1 \frac{C_{in}}{C_1}, \\ F_2 &= G_2 B_2 H_2 = G_2 \frac{C_{L,2}}{C_{in}} = G_2 H_2 \frac{C_{in}}{C_2}. \end{aligned}$$

Assume that leg 1 has n stages and leg 2 has m stages. Then, equalizing the leg delays to minimize the delay of the branching circuit yields the constraint:

$$\hat{D}_1(n) = n\left(\frac{G_1 H_1 C_{in}}{C_1}\right)^{\frac{1}{n}} + P_1 \quad \stackrel{!}{=} \quad m\left(\frac{G_2 H_2 C_{in}}{C_2}\right)^{\frac{1}{m}} + P_2 = \hat{D}_2(m).$$

In the special case for legs with equal lengths, $n = m$, and equal path parasitic delays, $P_1 = P_2$, this constraint simplifies to

$$\frac{C_1}{C_2} = \frac{G_1 H_1}{G_2 H_2}. \tag{10}$$

Then, the branch constraint, $C_{in} = C_1 + C_2$, enables us to size the stage-1 gates of the legs such that the input capacitances are

$$C_1 = \frac{G_1 H_1}{G_1 H_1 + G_2 H_2} C_{in}, \qquad C_2 = \frac{G_2 H_2}{G_1 H_1 + G_2 H_2} C_{in}.$$

The equivalent inverter path of this branching circuit has load capacitance $C_L = (G_1 H_1 + G_2 H_2) C_{in}$.

In case where both legs have the same number of stages but different parasitic delays, we may assume that Equation **(10)** holds approximately:

$$\frac{C_1}{C_2} \approx \frac{G_1 H_1}{G_2 H_2}. \tag{11}$$

If the number of stages differ, we may append inverter pairs to the shorter leg until the path lengths differ by at most one stage. Then, Approximation **(11)** may still be accurate enough to permit path sizing of the equivalent inverter path with load $C_L = (G_1 H_1 + G_2 H_2) C_{in}$. If the best number of stages requires increasing the leg lengths, we can accomplish the task by appending inverter pairs. Otherwise, if the best path length is smaller, we may redesign the logic on the legs to reduce the number of stages or settle with a suboptimal design.

■ **Example 3.14:** Branch Design with Logic

We wish to minimize the delay of the the branching circuit in **Figure 3.14** with path electrical efforts $H_1 = 9$ and $H_2 = 15$.

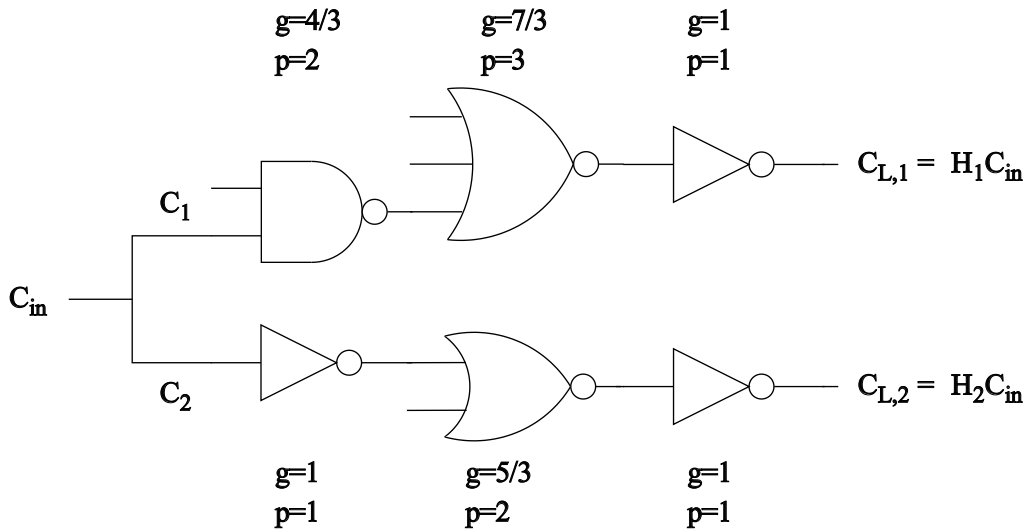


Figure 3.14: A branching circuit with logic gates on its legs.

We begin by deriving the path logical efforts and path parasitic delays of the legs. **Figure 3.14** annotates each gate with its logical effort and parasitic delay. The path logical efforts of the legs are the products of their gate logical efforts:

$$G_1 = \frac{4}{3} \cdot \frac{7}{3} \cdot 1 = \frac{28}{9}, \qquad G_2 = 1 \cdot \frac{5}{3} \cdot 1 = \frac{5}{3}.$$

The path parasitic delays of the legs are the sums of their gate parasitic delays:

$$P_1 = 2 + 3 + 1 = 6, \qquad P_2 = 1 + 2 + 1 = 4.$$

We notice that both legs have 3 stages but different path parasitic delays. The equivalent inverter path enables us to determine the best number of stages approximately. The equivalent inverter path has input capacitance C_{in} and load capacitance

$$C_L = (G_1 H_1 + G_2 H_2) C_{in} = \left(\frac{28}{9} \cdot 9 + \frac{5}{3} \cdot 15\right) C_{in} = 53 C_{in}.$$

Thus, the equivalent inverter path has path effort $F = C_L/C_{in} = 53,$ for which the **path sizing calculator** prescribes a best number of stages of $n = 3.$ Since both legs of the branching circuit have 3 stages already, we retain the logic without modifications. To minimize the delay, Approximation **(11)** permits estimating the input capacitances of the stage-1 gates of the legs as:

$$C_1 \approx \frac{G_1 H_1}{F} C_{in} = \frac{28}{53} C_{in}\,, \qquad C_2 \approx \frac{G_2 H_2}{F} C_{in} = \frac{25}{53} C_{in}\,.$$

Furthermore, if each of the gates bears stage effort $\hat{f} \approx \sqrt[n]{F} = \sqrt[3]{53} = 3.765,$ then the path effort delay of the branching circuit is approximately $3\hat{f} = 11.3.$ Including the path parasitic delays, we estimate the delay of leg 1 to be $\hat{D}_1(3) \approx 3\hat{f} + P_1 = 17.3$ and the delay of leg 2 to be $\hat{D}_2(3) \approx 3\hat{f} + P_2 = 15.3.$ This approximation is easy to obtain on the back of an envelope.

Alternatively, with a little extra work, we apply the exact **1d-analysis** method after determining path size $n = 3$ with the approximating equivalent inverter path. We find parasitic delay compensation $\Delta p = 2,$ and assign effort delay $d/3$ to each gate of leg 1 and $(d+2)/3$ to each gate of leg 2. We obtain the polynomial $d^3(d+2)^3 - 27 \cdot 28(d+2)^3 - 25 \cdot 27 d^3,$ which we do not need to expand further for **WolframAlpha** to find the roots. The only positive real root is $d = 10.495,$ and the resulting minimal circuit delay $\hat{D} = 16.5$ delay units. Note that $\hat{D} = 16.5$ is reasonably close to our approximate leg delays of $\hat{D}_1(3) = 17.3$ and $\hat{D}_2(3) = 15.3,$ validating the accuracy of the approximation. The 1d-analysis rewards the extra design effort with exact rather than approximate information for gate sizing, by prescribing different stage efforts $\hat{f}_1 = d/3 = 3.5$ for leg 1 and $\hat{f}_2 = (d+2)/3 = 4.2$ for leg 2. With a little practice and the aid of mathematical software like **WolframAlpha** for root finding, the 1d-analysis is as easy as the back-of-the-envelope approximation.

■ **Example 3.15:** Branch Design with Path Sizing

We wish to minimize the delay of the branching circuit in **Figure 3.15** with path electrical efforts $H_1 = H_2 = 15.$

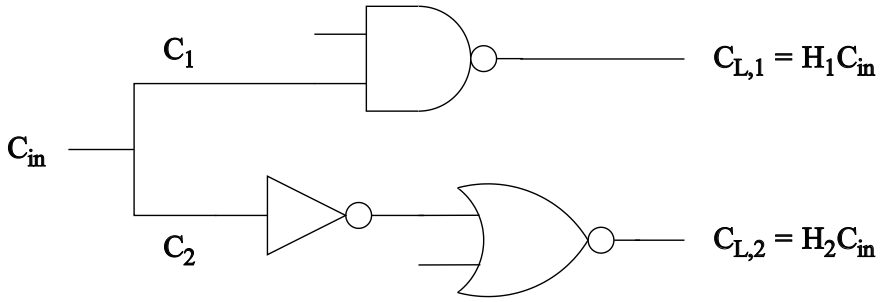


Figure 3.15: A branching circuit with logic gates on the legs.

Analogous to **Example 3.14** we first derive the approximating equivalent inverter path to determine the best path size. The leg logical effort of leg 1 is the logical effort of the NAND gate, $G_1 = 4/3$, and for leg 2, we obtain $G_2 = 1 \cdot 5/3 = 5/3$. The load capacitance of the approximating equivalent inverter path is

$$C_L = (G_1 H_1 + G_2 H_2) C_{in} = \left(\frac{9}{3} \cdot 15\right) C_{in} = 45 C_{in} \,.$$

For path effort $F = C_L / C_{in} = 45$, the **path sizing calculator** prescribes a path length of $n = 3$. Since leg 1 of our branch has only one stage, we append two inverters. Leg 2 we leave unmodified, because two stages may be sufficiently fast, and appending an inverter pair would overshoot the best number of stages by one. The modified branching circuit is shown in **Figure 3.16**.

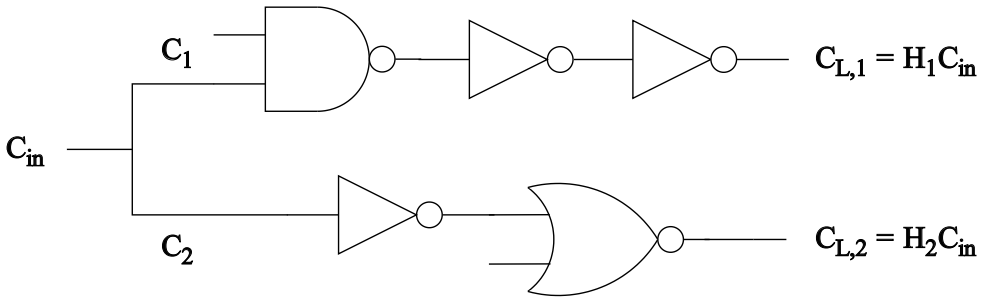


Figure 3.16: Branching circuit with leg 1 extended for best leg length.

Next, we apply the ***1d-analysis method*** to minimize the branch delay. The path parasitic delays are $P_1 = 2 + 1 + 1 = 4$ for leg 1 and $P_2 = 1 + 2 = 3$ for leg 2. We find a parasitic delay compensation of $\Delta p = 1$, and assign stage delay $d/3$ to each gate of leg 1 and $(d+1)/2$ to each gate of leg 2. Multiplying the effort delays of each path, we obtain $(d/3)^3 = G_1 H_1 C_{in} / C_1$ and $((d+1)/2)^2 = G_2 H_2 C_{in} / C_2$. Substituting C_1 and C_2 in the branch constraint, we obtain the polynomial $d^3(d+1)^2 - 4 \cdot 25 d^3 - 20 \cdot 27 (d+1)^2$. **WolframAlpha** finds the positive real root $d = 11.5$. If each gate in leg 1 bears stage effort $\hat{f}_1 = d/3 = 3.8$ and each gate in leg 2 stage effort $\hat{f}_2 = (d+1)/2 = 6.2$, then the branching circuit has a minimum delay of $\hat{D} = 15.5$ delay units. Note that the stage effort $\hat{f}_2$ exceeds the best stage effort $f^\star = 3.59$ by more than $70\,\%$. To reduce stage effort $\hat{f}_2$, we may want to try extending leg 2 with an inverter pair after all.

3.6.2. Branches with Driver Path

When we design a circuit with a branch, the input capacitance of the branch presents a load to another logic circuit. In the following, we discuss the design of circuits with branches that are driven by a logic path. **Figure 3.17** shows an example, where the driver path consists of a single stage only, a NOR gate.

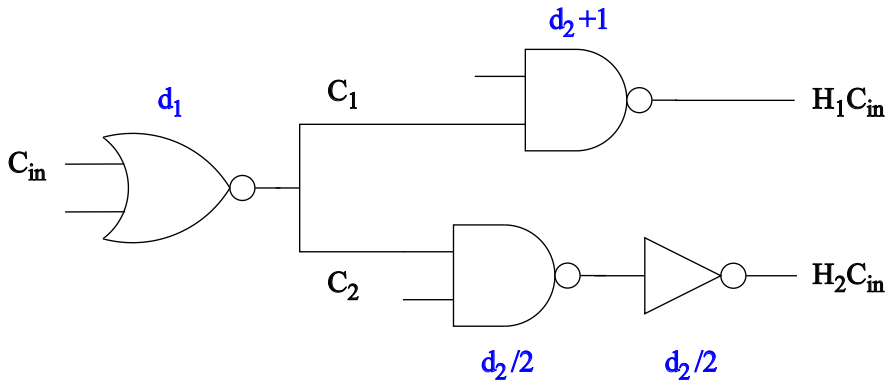


Figure 3.17: Branching circuit with driver path.

If we wish to minimize the delay of the branching circuit in **Figure 3.17**, we are faced with the problem of how much effort the driver path and how much effort the legs of the branch should bear. In general, the legs of the branch may bear different path efforts, e.g. d_2 and $d_2 + 1$ in **Figure 3.17**, and, furthermore, the path effort of the driver path may differ from path efforts of the legs. We know how to use the **1d-analysis method** to assign effort delays to the gates of the legs. Assume the driver path bears effort delay d_1, and the branch bears effort delay d_2, which we allocate to the legs according to the **1d-analysis method**. **Figure 3.17** includes the allocation of the effort delays to the gates of the circuit. Now, the design task involves deriving effort delays d_1 and d_2 such that the circuit delay is minimized. The *2d-analysis method* enables us to find these two delays:

2d-Analysis Method

Given a branching circuit and a driver path with input capacitance C_{in} and load capacitances H_1C_{in} and H_2C_{in}. Follow these steps to minimize the circuit delay:

1. Given an n-stage driver path, assign effort delay d_1/n to each gate of the driver path, and use the **1d-analysis** method to allocate delay d_2 to the gates of the legs.
2. Form the products of the leg efforts and the driver path efforts, and use the branch constraint to express d_1 as a function of d_2 or vice versa.
3. Express path effort delay $D = d_1 + d_2$ as a function of d_2 (or d_1), and minimize D, e.g. by means of calculus.

We introduce the **2d-analysis** method with two examples.

■ **Example 3.16:** Branch Design with 2d-Analysis

We minimize the delay of the circuit in **Figure 3.17** using the **2d-analysis** method. Step 1 of the **2d-analysis** method, allocating effort delay d_1 to the driver path and d_2 to the gates of the branch, is already complete. Note that the **1d-analysis** method assigns path effort delay $d_2 + 1$ to leg 1, which is by one delay unit larger than path effort delay $d_2/2 + d_2/2 = d_2$ of leg 2 in order to compensate for the larger parasitic delay of leg 2. Next, in step 2 of the **2d-analysis**, we derive the telescoping products of the leg efforts:

$$\begin{aligned} d_2 + 1 &= g_{nand} \frac{H_1 C_{in}}{C_1} = \frac{4 H_1 C_{in}}{3 C_1}, \\ \left(\frac{d_2}{2}\right)^2 &= g_{nand} g_{inv} \frac{H_2 C_{in}}{C_2} = \frac{4 H_2 C_{in}}{3 C_2}. \end{aligned}$$

Rearranging these equations yields expressions for leg input capacitances C_1 and C_2:

$$C_1 = \frac{4 H_1 C_{in}}{3(d_2 + 1)}, \qquad C_2 = \frac{4 H_2 C_{in}}{3(d_2/2)^2}.$$

The effort delay of the driver stage with load capacitance $C_1 + C_2$ is:

$$d_1 = g_{nor} \frac{C_1 + C_2}{C_{in}} = \frac{5}{3} \frac{C_1 + C_2}{C_{in}}.$$

Here, we have applied the branch constraint that the branch input capacitance, i.e. the load of the driver path, is the sum of leg input capacitances. Substituting C_1 and C_2 in this equation, yields d_1 as a function of d_2:

$$d_1 = \frac{5}{3} \frac{\frac{4 H_1 C_{in}}{3(d_2+1)} + \frac{4 H_2 C_{in}}{3(d_2/2)^2}}{C_{in}} = \frac{20 H_1}{9(d_2 + 1)} + \frac{80 H_2}{9 d_2^2}.$$

In step 3 of the **2d-analysis** we express path effort delay D of the whole circuit as a function of d_2:

$$D = d_1 + d_2 = \frac{20}{9} \frac{H_1}{d_2 + 1} + \frac{80}{9} \frac{H_2}{d_2^2} + d_2.$$

To find the minimum of path effort delay D, with take derivative w.r.t d_2, and set the result to zero:

$$\frac{\partial D}{\partial d_2} = \frac{20}{9} \frac{-H_1}{(d_2 + 1)^2} + \frac{80}{9} \frac{-2 H_2}{d_2^3} + 1 \stackrel{!}{=} 0.$$

Rearranging this equation into polynomial form reduces our minimization problem into yet another application of root finding:

$$d_2^3(d_2+1)^2 - \frac{20}{9}H_1 d_2^3 - \frac{160}{9}H_2(d_2+1)^2 = 0\,.$$

In principle, we also need to check whether the 2^{nd} derivative is positive for $D(d_2)$ to attain a minimum. Alternatively, we can inspect the plot of $D(d_2)$ in **WolframAlpha** to verify a minimum at the relevant root. For instance, given path electrical efforts $H_1 = H_2 = 15,$ we invoke **WolframAlpha** to find the roots of $d_2^3(d_2+1)^2 - (100/3)d_2^3 - (800/3)(d_2+1)^2.$ The only positive real root is $d_2 = 7.8.$ Thus, we find $d_1 = 6,$ and the minimum path effort delay of the branching circuit is $D = 13.8.$ Including path parasitic delay $P_2 = 5,$ the minimum path delay of the entire circuit is $\hat{D} = D + P_2 = 18.8$ delay units.

■ **Example 3.17:** Fork Design with 2d-Analysis

In Section **1d-Analysis Method**, we minimize the delay of the 3-fork in **Figure 3.10**. For leg electrical efforts $H_1 = H_2 = 25,$ we obtain a minimum delay of $\hat{D} = 14.91$ units. In this example, we investigate whether it is beneficial to replace two inverters of the 3-fork with one inverter on the drive path of a 2-fork. **Figure 3.18** shows the circuit, which is logically equivalent to a 3-fork, but saves one inverter.

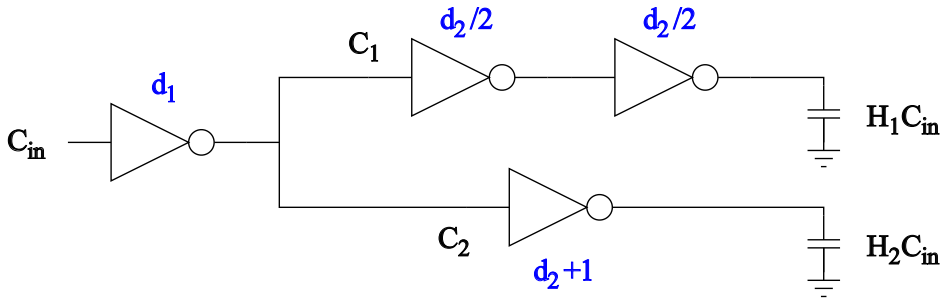


Figure 3.18: A 2-fork with a driving inverter is logically equivalent to the 3-fork in **Figure 3.10**.

We use the **2d-analysis** method to minimize the delay of the 2-fork with a 1-stage driver path. In step 1 we allocate the effort delays as annotated in **Figure 3.18** already. The inverter in lower leg 2 shoulders parasitic delay compensation $\Delta p = P_1 - P_2 = 2 - 1 = 1$ w.r.t. upper leg 1. Next, we derive the telescoping products of the effort delays for the legs of the 2-fork:

$$\left(\frac{d_2}{2}\right)^2 = \frac{H_1 C_{in}}{C_1}\,, \qquad d_2 + 1 = \frac{H_2 C_{in}}{C_2}\,,$$

and for the driver path with load capacitance $C_1 + C_2$:

$$d_1 = \frac{C_1 + C_2}{C_{in}}\,.$$

Substituting C_1 and C_2 yields d_1 as a function of d_2:

$$d_1 = \frac{4H_1}{d_2^2} + \frac{H_2}{d_2+1}\,.$$

Step 3 minimizes path effort delay $D = d_1 + d_2$, by taking the derivative of D w.r.t. d_2, and finding the relevant root of the resulting polynomial:

$$d_2^3(d_2+1)^2 - H_2 d_2^3 - 8H_1(d_2+1)^2\,.$$

For $H_1 = H_2 = 25$, WolframAlpha finds the positive real root $d_2 = 6.9$, from which we deduce $d_1 = 5.2$. Including parasitic delay $P_1 = 3$, the minimum path delay is $\hat{D} = d_1 + d_2 + P_1 = 15.1$ units, which is marginally larger than the minimum delay of the 3-fork.

3.6.3. Reconvergent Branches

We conclude our discussion of branching circuits by applying the 1d-analysis and 2d-analysis methods to branches whose legs reconverge at the inputs of a gate. Figure 3.19 shows such a circuit. The NAND gate in stage 1 drives a branch with legs in stage 2 that reconverge at the NAND gate in stage 3. Furthermore, each of the inputs of the circuit branches. One of the two legs drives the stage-1 NAND gate and the other the stage-2 NAND gate. The design challenge is to minimize the delay of this circuit.

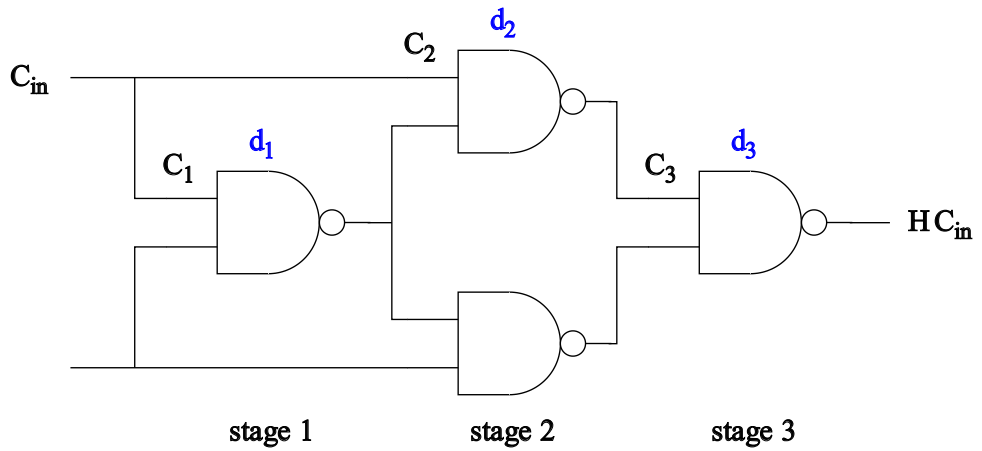


Figure 3.19: A circuit with reconverging branches.

First, we recognize that the longest path through the circuit traverses three NAND gates, one in each stage. The circuit is symmetric if we assume that the NAND gates in stage 2 have equal size. Then, the path from the upper input through the stage-1 NAND gate, the upper stage-2 NAND gate, and the NAND gate in stage 3 is representative for the four longest paths from either input to the output. This is our path of interest.

To determine the delay of the circuit, we allocate effort delays d_1, d_2, and d_3 to the corresponding stages, as shown in **Figure 3.19**. Analysis of the circuit reveals the stage effort delays:

$$\begin{aligned} d_1 &= g_{nand}\frac{C_2 + C_2}{C_1} = \frac{8}{3}\frac{C_2}{C_1}, \\ d_2 &= g_{nand}\frac{C_3}{C_2} = \frac{4}{3}\frac{C_3}{C_2}, \\ d_3 &= g_{nand}\frac{HC_{in}}{C_3} = \frac{4}{3}\frac{HC_{in}}{C_3}. \end{aligned}$$

Here, the load of the stage-1 NAND gate is the sum of the leg input capacitances, both of which are C_2 assuming that the stage-2 NAND gates have equal size. The branch constraint for the circuit input is

$$C_{in} = C_1 + C_2,$$

enabling us to capture the relation between the effort delays in a single equation:

$$d_1 d_2 d_3 - \left(\frac{4}{3}\right)^2 H d_1 - 2\left(\frac{4}{3}\right)^3 H = 0. \tag{12}$$

To solve Equation **(12)**, we need to constrain variables d_1, d_2, and d_3. A simplistic constraint assumes that all stage efforts are equal, $d_1 = d_2 = d_3$, as motivated by the key insight from the analysis of multistage paths. Although our circuit is not a simple path, this constraint yields a univariate polynomial of degree three. For example, given $H = 6$, we obtain the positive real root $d_1 = 4.18$ and path delay $D_{1d} = 3d_1 + 3p_{nand} = 18.54$.

We might expect a better design from a **2d-analysis**, assuming that the NAND gates in stages 2 and 3 are the legs of the branch, with the additional constraint that the stage-3 NAND gate belongs to both legs. To equalize the leg delays, we allocate effort delay $d_2 = d_3$ to stages 2 and 3. However, we assume a potentially different delay d_1 for the driver path in stage 1. Now we wish to minimize path delay $D = d_1 + 2d_2$. We rearrange Equation **(12)** to express d_1 as a function of d_2, assuming $d_3 = d_2$:

$$d_1 = \frac{2\left(\frac{4}{3}\right)^3 H}{d_2^2 - \left(\frac{4}{3}\right)^2 H},$$

such that

$$D = \frac{2\left(\frac{4}{3}\right)^3 H}{d_2^2 - \left(\frac{4}{3}\right)^2 H} + 2d_2\,.$$

We minimize D by setting the derivative of D w.r.t. d_2 to zero, and find the root $d_2 = 4.7$ for $H = 6.$ As a result, we obtain $d_1 = 2.5$ and $D_{2d} = d_1 + 2d_2 + 3p_{nand} = 17.89.$ We conclude that the 2d-analysis produces a slightly faster circuit than the 1d-analysis.

To reduce the delay of the circuit beyond $D_{2d},$ we observe that the branches at the circuit inputs have the topology of reconvergent 1-forks that drive the stage-2 gates. One leg includes the stage-1 NAND gate instead of an inverter, and the other leg contains no logic. In Section **1d-Analysis Method**, we identified a 1-fork as an undesirable circuit. As a case in point, we improve the circuit in **Figure 3.19** by replacing the 1-fork with 2-fork topologies. To that end we insert back-to-back inverters as shown in **Figure 3.20**. The back-to-back inverters in the upper leg and the NAND gate in the lower leg resemble a 2-fork, except that the 1-stage leg contains the NAND gate rather than an inverter.

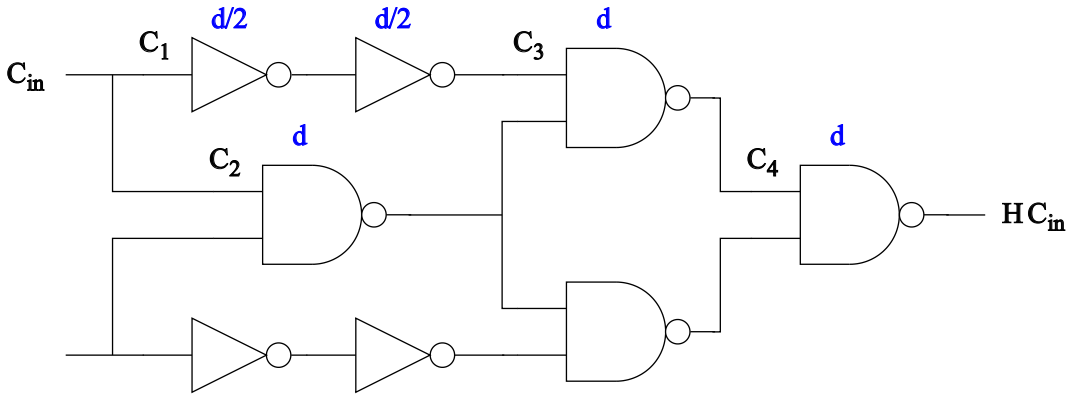


Figure 3.20: Improved version of reconverging branch circuit in **Figure 3.19**.

Since the path parasitic delay of two inverters equals the parasitic delay of a NAND gate, the delays of the two legs of the 2-forks are equal if we allocate effort delay d to the NAND gate and $d/2$ to each of the inverters. Using a **1d-analysis**, we also allocate effort delay d to the remaining NAND gates, as shown in **Figure 3.20**. The stage efforts of the inverter pair and the NAND gates are:

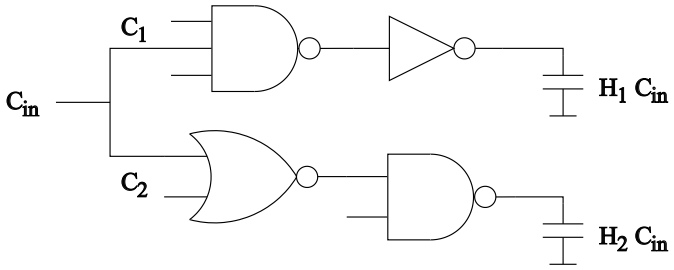
$$\left(\frac{d}{2}\right)^2 = \frac{C_3}{C_1}\,, \qquad d = \frac{4}{3}\frac{2C_3}{C_2}\,, \qquad d = \frac{4}{3}\frac{C_4}{C_3}\,, \qquad d = \frac{4}{3}\frac{HC_{in}}{C_4}\,.$$

Applying branch constraint $C_{in} = C_1 + C_2$ yields:

$$d^4 - 2\left(\frac{4}{3}\right)^3 Hd - 4\left(\frac{4}{3}\right)^2 H = 0\,.$$

Given $H = 6$, we find the positive real root $d = 3.44$. The minimum delay of the circuit in **Figure 3.20** is $\hat{D}_{1d} = 3d + 3p_{nand} = 16.3$ units, which is faster than the original circuit in **Figure 3.19**.

Use the ***1d-analysis method*** to minimize the delay of the branching circuit assuming $H_1 = H_2 = 12$.



a. Determine the leg parasitic delays and allocate effort delays using one parameter d.
b. For each leg, form the products of the effort delays, and derive a polynomial in d.
c. Determine the root of the polynomial and the minimum leg delays $\hat{D}_1$ and $\hat{D}_2$.

The branching circuit has a 3-input NAND gate followed by an inverter on leg 1 and a 2-input NOR gate followed by a 2-input NAND gate on leg 2. The path parasitic delay of leg 1 is

$$P_1 = p_{nand3} + p_{inv} = 3 + 1 = 4\,,$$

and of leg 2

$$P_2 = p_{nor2} + p_{nor2} = 2 + 2 = 4\,.$$

Since $\Delta p = P_1 - P_2 = 0$, we do not need a parasitic delay compensation. Therefore, we assume that each leg has effort delay d, and allocate stage effort delay $d/2$ to each of the four gates.

The product of the effort delays of leg 1 equals its path effort including the path logical effort:

$$\left(\frac{d}{2}\right)^2 = \frac{G_1 H_1 C_{in}}{C_1}\,,$$

and for leg 2:

$$\left(\frac{d}{2}\right)^2 = \frac{G_2 H_2 C_{in}}{C_2}\,.$$

The path logical effort of leg 1 is $G_1 = g_{nand3}\, g_{inv} = 5/3$. Given $H_1 = 12$, we find $G_1 H_1 = 20$. The product equation yields an expression for

$$C_1 = \frac{80}{d^2} C_{in}\,.$$

Analogously, for leg 2 we find path logical effort $G_2 = g_{nor2}\, g_{nand2} = 20/9$, and with $H_2 = 12$, $G_2 H_2 = 80/3$. Rearranging the product equation results in

$$C_2 = \frac{320}{3d^2} C_{in}\,.$$

Next, we substitute C_1 and C_2 in the branch constraint to obtain a polynomial in d:

$$\begin{aligned} C_{in} &= C_1 + C_2 \\ C_{in} &= \frac{80}{d^2} C_{in} + \frac{320}{3d^2} C_{in} \\ d^2 &= 560/3\,. \end{aligned}$$

This equation represents a degraded polynomial of degree 2, which is easy to solve. Since we are interested in positive delays only, we find effort delay d by taking the positive square root:

$$d = \sqrt{\frac{560}{3}} = 13.66\,.$$

Therefore, both legs have a minimum path delay of $\hat{D}_1 = d + P_1 = \hat{D}_2 = d + P_2 = 17.66$ time units.

Hide

3.7. Summary

The method of logical effort enables us to design digital circuits with minimum delay. It offers insight, recipies for systematic delay analysis, and methods for gate and path sizing. The *1d-analysis* and *2d-analysis* methods facilitate the design of forks and branching circuits, that rely on the delay minimization of a basic multistage path. The following steps summarize the path design procedure:

1. Calculate the path effort $F = GBH$ of the path of interest. Path logical effort G is the product of the gate logical efforts, and path electrical effort H is the telescoping product of the gate fanouts. For paths without branches, path branching effort B equals 1.
2. Use the **path sizing calculator** to determine the best number of stages $n^\star$ for the path of interest. If necessary, adapt the circuit to have $n^\star$ stages.
3. Size the gates on the path of interest such that each gate bears stage effort $\hat{f} = F^{1/n^\star}$.
4. Calculate path parasitic delay P as the sum of the gate parasitic delays. The minimum path delay is $\hat{D} = n^\star \hat{f} + P$.

The key insight due to the path design procedure is the fact that each stage must bear the same effort to minimize the path delay. Together with the insight that the legs of a branch must have equal delay to minimize the delay of the whole branching circuit, the method of logical effort leads to the 1d and 2d-analysis methods. These methods permit the design of complex circuit topologies with nothing but paper and pencil, and the aid of a root finding package.