Exercise 1

The two signal flow graphs in Figure 1 realizes the same function. The delay for a multiplication and an addition is $4t_d$ and $2t_d$, respectively. The input signal $x(n)$ must be held at constant value $x(j)$ during a certain time, before the new value $x(j+1)$ is allowed to occur at the input. The theoretical minimal time (neglecting propagation delay through delay elements) between the values of $x(j)$ and $x(j+1)$ is

$$T_{min} = \max_i \left\{ t_i / N_i \right\}$$

where T_i is the total delay due to the arithmetic operations, and N_i is the number of delay elements in the directed loop i.

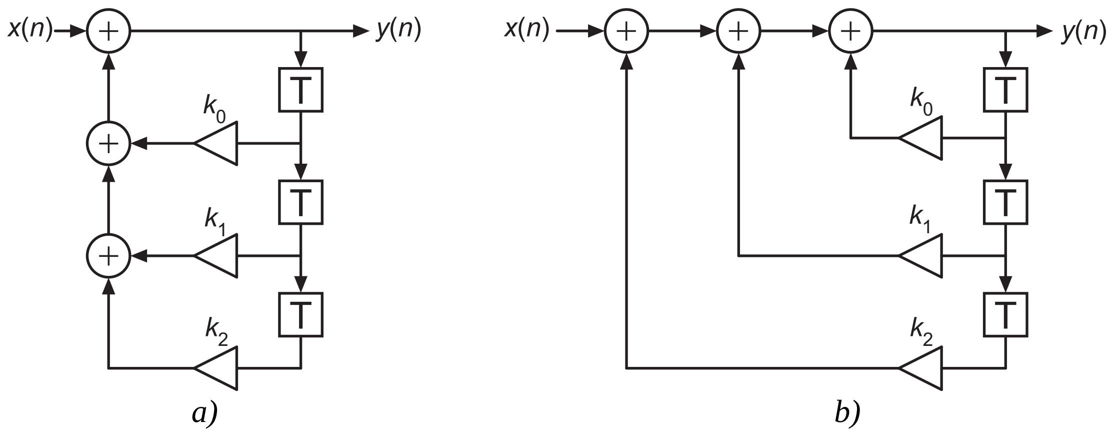


Figure 1. Two different signal flow graphs of the same function.

a) Determine $T_{min(a)}$ and $T_{min(b)}$. Neglect the propagation delay through the delay elements.

b) Which signal flow graph results in the fastest circuit?

c) The lowest possible power supply voltage of the circuit realized from Figure 1a) is 2.0 V for a given demand of performance (throughput). Determine the lowest possible power supply voltage of the circuit realized from Figure 1b) when $V_t = 0.35\text{ V}$ and $t_d \propto V_{dd} \big/ (V_{dd} - V_t)^2$. Both circuits should have their delay elements placed as in Fig. 1.

d) Determine $P_{(b)} / P_{(a)}$.

Exercise 2

a) Loop unroll the signal flow graph in Figure 2 and use arithmetic transformations to shorten the critical path.

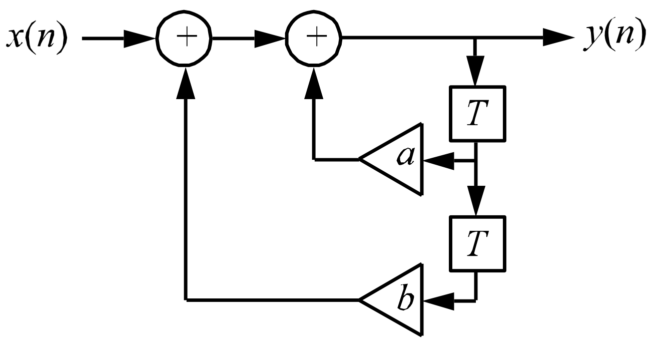


Figure 2. Signal flow graph.

b) What are the advantages of unrolling?

Exercise 3

In Figure 3, a shift register is shown. The number of registers is 64 and the clock frequency is 1.00 GHz. The D flip-flops are all positive edge triggered. The set up time t_{su} and the hold time t_h for the D flip-flops are 0.17 ns and 0.05 ns, respectively. The new data is propagated to the output 0.14 ns after the rising clock edge (i.e. t_{cq} = 0.14 ns). V_{dd} = 2.5 V. V_t = 0.30 V. The propagation delay in the chosen process technology is in proportion to $V_{dd}/(V_{dd}-V_t)^{1.5}$.

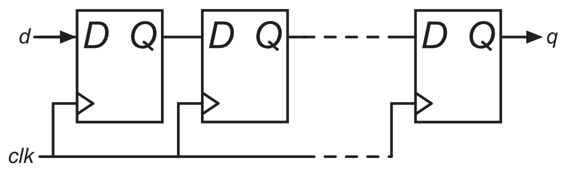


Figure 3. Shift register.

a) Determine the lowest possible power supply voltage for the original circuit. How large is the relative power consumption?

b) Introduce interleaving in the shift register (2x32). You may use one multiplexer and two clock signals. The throughput should be the same as in the original circuit. Assume that the data stream $d = [d_0, d_1, d_2, d_3, ...]$ can be divided into two data streams $d_a = [d_0, d_2, d_4, ...]$ and $d_b = [d_1, d_3, d_5, ...]$ (How?). Sketch the circuit and the waveforms. Determine the relative power consumption using the same supply voltage as in a) compared with the original voltage. Neglect the propagation delay and the power consumption of the multiplexer.

c) Determine the lowest possible power supply voltage of the interleaved circuit. How large is the relative power consumption compared with the original circuit?

Exercise 4

A four-input multiplexer realized by three two-input multiplexers is shown in Figure 4. The inputs are periodically selected in the order n_0, n_1, n_2, n_3. A new input is selected in each clock cycle. The propagation delay of one multiplexer is t_d.

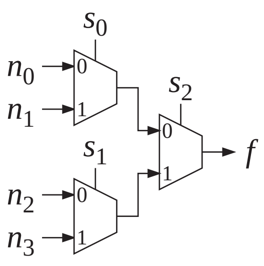


Figure 4. Four-input multiplexer.

a) Determine the select signals s_0, s_1, s_2 and calculate the propagation delay.

b) If the input signal n_j (j = 0, 1, 2, 3) always is valid a certain time $t_n > t_d$ before it is selected, a faster circuit can be designed with the same hardware. Sketch the new circuit and the select signals s_0, s_1, and s_2. Determine the propagation delay from the time when the select signals become valid.

c) How can the result in b) be used to save power?

Exercise 5

a) In Figure 5 a multiplier is shown. The input signal x is multiplied with 0.375. How many inputs of the adder is connected to the sign bit x_0?

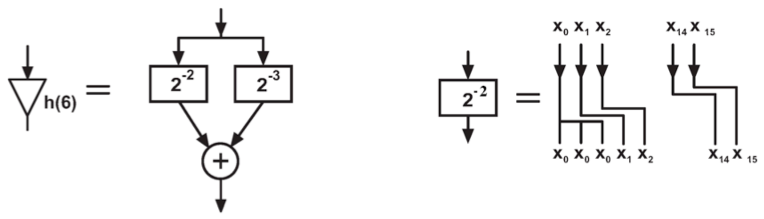


Figure 5. Multiplier realized by two hard wire shifts and one adder.

b) Show that the result shown in Figure 6 always is true.

$$\begin{array}{cccccccccccc}
1 & 1 & 1 & 1 & \overline{a}_0 & a_1 & a_2 & a_3 & - - - & a_{14} & a_{15} \\
 & & & + & 1 & & & & & & \\
\hline
a_0 & a_0 & a_0 & a_0 & a_0 & a_1 & a_2 & a_3 & - - - & a_{14} & a_{15}
\end{array}$$

Figure 6. Interesting result of two's complement addition.

c) How is it possible to save power with the use of the result in b)?

Exercise 6

An N-times-N matrix A should be calculated. Each element in A is calculated as shown in the equation below where $c_1(x, i)$ and $c_2(y, j)$ are constants and $B(i, j)$ are variables.

$$A(x, y) = \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} c_1(x, i) c_2(y, j) B(i, j)$$

a) How many multiplications are required to calculate A?

b) How many multiplications are required if $c_1(x, i)\cdot c_2(y, j)$ are precalculated to a new constant $C(x, y, i, j)$?

c) Rewrite the equations so that the number of multiplications is reduced.

d) Determine the number of memory access in b) and c).

Exercise 7

The algorithm shown in Figure 7 accumulates an even number of samples N by adding the samples in a sequence $y(n) = x(n)+y(n–1)$; $n = 0,1, …, N–1$. The register is implicitly reset at $n = 0$. The algorithm should be optimized for low energy consumption using algorithm transformations and supply voltage scaling. Assume a constant capacitive load of adders and registers in implementations, and that the adder and the register hardware operate at a common supply voltage in an implementation. In a direct implementation of the algorithm in Figure 7 the supply voltage is $V_{dd0} = 1.8$ V after minimization with respect to the throughput requirement.

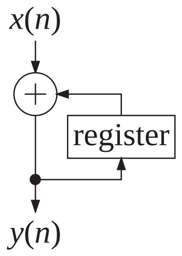


Figure 7. Signal-flow graph of an accumulate algorithm.

a) How large energy savings can be obtained with direct pipelining or interleaving of the algorithm?

b) Transform the algorithm using loop unrolling such that two samples are processed concurrently. Hence the new algorithm should perform the operations $y(2n) = x(2n)+y(2n–1)$ and $y(2n+1) = x(2n+1)+y(2n)$; $n = 0,1, …, N/2–1$. Draw the signal-flow graph of the new algorithm.

c) Use the associative and distributed arithmetic laws to reduce the latency of the critical loop of the algorithm in b). The final latency should equal the sum of one two-input adder and one register delay. Draw the signal-flow graph of the new algorithm.

d) Pipeline the algorithm in c) with as few registers as possible so that the throughput becomes limited by the critical loop.

e) Assume the same supply voltage V_{dd0} is used in direct implementations of the original algorithm in Figure 7 and the resulting algorithm of d). Estimate the relative energy consumption of the two implementations.

f) Minimize the energy consumption of resulting algorithm of d) by scaling the supply voltage. Assume the propagation delay t_p is proportional to $V_{dd}/(V_{dd}–V_t)^{1.5}$ where $V_t = 0.45$ V. Estimate the relative energy savings after supply voltage scaling compared with a direct implementation of the original algorithm in Figure 7.

Solution 1

a)

$$T_{min(a)} = max\left\{\frac{T_0}{N_0}, \frac{T_1}{N_1}, \frac{T_2}{N_2}\right\} = max\left\{\frac{2T_{add} + T_{mult}}{1}, \frac{3T_{add} + T_{mult}}{2}, \frac{3T_{add} + T_{mult}}{3}\right\}$$

$$= 2T_{add} + T_{mult} = 8t_d$$

$$T_{min(b)} = max\left\{\frac{T_0}{N_0}, \frac{T_1}{N_1}, \frac{T_2}{N_2}\right\} = max\left\{\frac{T_{add} + T_{mult}}{1}, \frac{2T_{add} + T_{mult}}{2}, \frac{3T_{add} + T_{mult}}{3}\right\}$$

$$= T_{add} + T_{mult} = 6t_d$$

b) The signal flow graph in Figure 1 b)

c) $T_{min(b)} = \frac{4}{3}T_{min(a)} \Rightarrow T_{clock(b)} = \frac{4}{3}T_{clock(a)}$

$$\Rightarrow \frac{V_{dd(b)}}{\left(V_{dd(b)} - V_t\right)^2} = \frac{4}{3}\frac{V_{dd(a)}}{\left(V_{dd(a)} - V_t\right)^2}$$

where $V_t = 0.35\text{V}$ and $V_{dd(a)} = 2.0\text{V}$.

Solving the equation results in:

$V_{dd(b)} = k \pm \sqrt{k^2 - V_t^2}$ where $k = V_t + \frac{3(V_{dd(a)} - V_t)^2}{8V_{dd(a)}} \approx 0.8605$.

Hence $V_{dd(b)} \approx 1.647 \approx 1.65\text{V}$

d) $V_{dd(b)}{}^2 / V_{dd(a)}{}^2 \approx 0.68$

Solution 2

a) $y(2n) = x(2n) + ay(2n-1) + by(2n-2)$ (I)

$y(2n-1) = x(2n-1) + ay(2n-2) + by(2n-3)$ (II)

Use (II) in (I)

$y(2n) = x(2n) + ax(2n-1) + a^2y(2n-2) + aby(2n-3) + by(2n-2)$

$y(2n) = x(2n) + ax(2n-1) + (a^2 + b)y(2n-2) + aby(2n-3)$ (III)

Use (II) and (III) and sketch the circuit.

b) The modified circuit process two data in each clock cycle. The critical path of the original circuit is (after the move of one delay element) one multiplication and one addition plus the delay of the register. The maximal throughput is $1/(t_{add} + t_{mult} + t_{reg})$.

If pipelining is used (where?) then the maximal througput is $2/(2t_{add} + t_{mult} + t_{reg})$. By using loop unrolling and algebraic transformations the time margin can be increased which may be used for a higher throughput or for a lower power supply voltage which may result in power savings.

Solution 3

a) $T_{clk} \geq t_{su} + t_{cq} = 0.31\text{ ns},\ T_{clk} = 1\text{ ns}$

$$1 = 0.31\left(\frac{V_{DD(new)}}{(V_{DD(new)} - V_{TH})^{1.5}} \cdot \frac{(V_{DD(old)} - V_{TH})^{1.5}}{V_{DD(old)}}\right) \Rightarrow V_{DD(new)} \approx 0.75\text{ V}$$

Power consumption: ~ 9 % of the original circuit

b) Assuming that the divided data streams has the same transition activity as the original and neglecting the energy consumed by the multiplexer:
Half clock frequency ⇒ power consumption: 4.5 % of the original circuit.

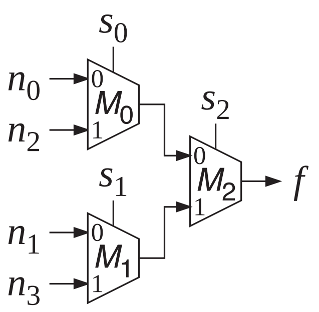
c) $V_{DD(new)} \approx 0.53\text{ V}$ ⇒ Power consumption: 2.2 % of the original circuit. Note that we have not considered any overhead of using two clocks and the power consumption of the multiplexer.

Solution 4

a) The table below shows how the select signals can be chosen ("-" = don't care). For example, when the input signal s_0 change value from 0 to 1, n_1 must propagate through two multiplexers before the output of the circuit is valid. Hence, the propagation time for the circuit is $2t_d$.

s_0	s_1	s_2	f
0	-	0	n_0
1	-	0	n_1
-	0	1	n_2
-	1	1	n_3

b) The figure below shows the four input multiplexer where two input signals have changed places. If the input signal n_j (j = 0, 1, 2, 3) always is valid a certain time $t_n > t_d$ before it is selected, the signal can propagate through the first multiplexer before it is selected. For example, consider the scenario where n_0 is selected after n_3. When n_3 is selected $s_0 = 0$, which means that n_0 is valid at the output of multiplexer M_0 before n_0 is selected. The "new" circuit results in a propagation delay of one multiplexer t_d. The extra time margin can be used for power supply scaling.



s_0	s_1	s_2	f
0	0	0	n_0
1	0	1	n_1
1	1	0	n_2
0	1	1	n_3

Solution 5

a) The sign bit x_0 is interconnected to 7 inputs of the adder while x_j (j>0) is connected to two inputs. This makes the sign bit the most capacitively loaded bit.

b) Numbers:

$$x = -a_0 2^{19} + \sum_{i=15}^{18} a_0 2^i + \sum_{j=0}^{14} a_{15-j} 2^j, y = -2^{19} + 2^{18} + 2^{17} + 2^{16} + (1-a_0)2^{15} + \sum_{j=0}^{14} a_{15-j} 2^j, z = 2^{15}$$

Show that sum is same:

$$y + z - x = -2^{19} + 2^{18} + 2^{17} + 2^{16} + 2 \cdot 2^{15} - 2^{15} a_0 + a_0 2^{19} - \sum_{i=15}^{18} a_0 2^i =$$

$$= 0 + a_0 \left(2^{19} - 2^{18} - 2^{17} - 2^{16} - 2 \cdot 2^{15}\right) = 0 + a_0 \cdot 0 = 0 \Rightarrow x = y + z$$

c) Do not sign extend, but invert the sign bit and add ones in the positions given by b). The ones should not consume dynamic power since they are not switching, and the logic nets can be simplified for the constant case.

Solution 6

a) $2N^4$

b) N^4

c) With the following manipulation of the equation, the number of multiplications will be reduced:

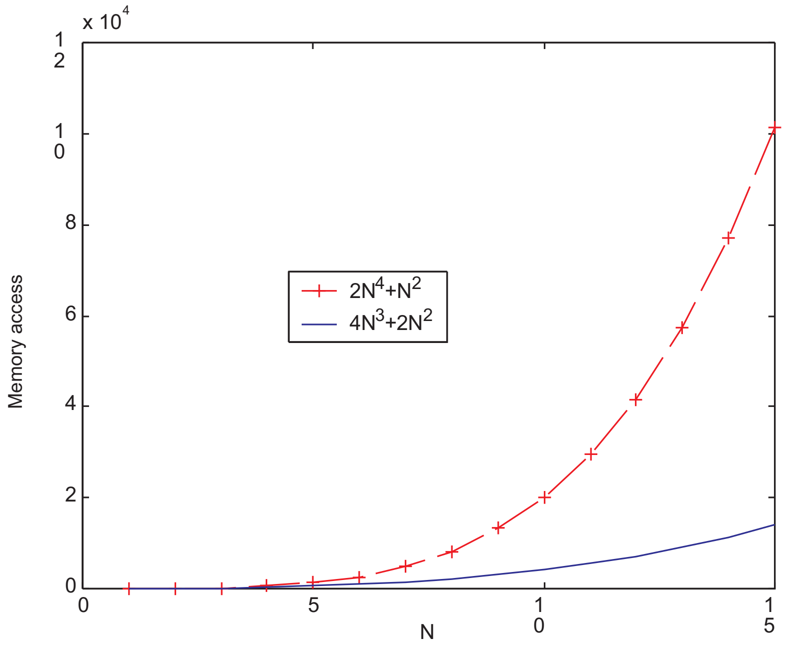
$$A(x, y) = \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} c_1(x, i) c_2(y, j) B(i, j) = \sum_{i=0}^{N-1} c_1(x, i) \sum_{j=0}^{N-1} c_2(y, j) B(i, j) \Rightarrow$$

$$A(x, y) = \sum_{i=0}^{N-1} c_1(x, i) Z(i, y), \qquad Z(i, y) = \sum_{j=0}^{N-1} c_2(y, j) B(i, j)$$

Calculating Z takes N^3 multiplications, N multiplications for one summation, and N^2 elements in Z gives totally N^3 multiplications. A also takes N^3 multiplications which gives a total number of $2N^3$ multiplications. The number of multiplications is reduced from N^4 to $2N^3$.

d) Memory access before the manipulation:
   $C - N^4$ (one memory access for every calculation)
   $B - N^4$ (one memory access for every calculation)
   $A - N^2$ (one memory access after one finished calculation)
   Total memory access = $N^2 + 2N^4$

   Memory access after the manipulation:
   $c_2 - N^3$, $B - N^3$, $Z - N^2$, $c_1 - N^3$, $Z - N^3$, $A - N^2$. Total memory access = $4N^3 + 2N^2$
   For N>2 the last equation uses least memory access.

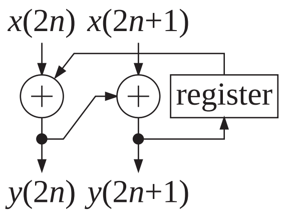


Memory access for both equations

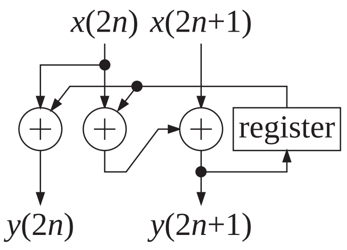
Solution 7

a) Pipelining and interleaving cannot be applied directly to a recursive algorithm, hence there is no energy to gain from these methods.

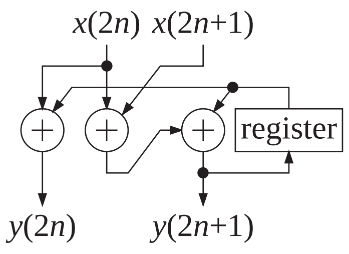
b) Signal-flow graph after loop unrolling:



c) To improve the latency of the critical loop, we need to transform operations out of the loop by reordering the additions. Start by simplifing the critical loop by moving the addition $x(2n) = x(2n)+y(2n)$ out of the loop. It can made by duplicating the left-most addition in b) according to the figure to the right.

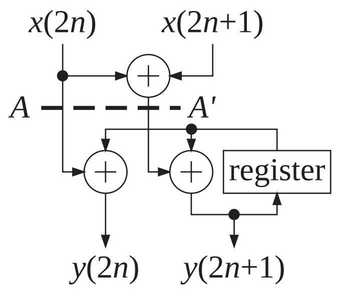


Reorder the additions such that the loop is shortened to one adder and one register. This is made in the figure to the right by swapping the order of adding $x(2n+1)$ and the register output.



This signal-flow graph has a critical loop with only one adder and one register in it and does hence solve our problem.

d) The critical path of the algorithm in c) goes through one register and two adders. This path can be reduced to one register and one adder by introducing pipelining in the sequential parts of the algorithm, obtaining the same latency of the critical path as that of the loop. A cut A—A' suitable for inserting two register to obtain the pipelining is indicated in the figure to the right.



e) Assume the energy consumption of a register operation to be E_{reg} and of an addition operation E_{add}. To perform an accumulation operation the algorithm in d) consumes the energy $E_d = N/2(3E_{reg}+3E_{add})=1.5N(E_{reg}+E_{add})$ while the original algorithm consumes $E_0 = N(E_{reg}+E_{add})$. Hence the algorithm in d) consumes 150% energy of the original algorithm if the same supply voltage is used.

f) The latency for computing an accumulation operation with an implementation based on the algorithm in d) is $t_1 = \frac{N}{2}kV_{dd1}(V_{dd1} - V_t)^{-1.5}$, where k is the proportionality constant, while an implementation based on the original algorithm has the latency $t_0 = NkV_{dd0}(V_{dd0} - V_t)^{-1.5}$. With the same throughput requirement,

$$t_1 = t_0 \Rightarrow \frac{V_{dd1}}{2(V_{dd1} - V_t)^{1.5}} = \frac{V_{dd0}}{(V_{dd0} - V_t)^{1.5}}.$$

Solving this equation numerically, we obtain $V_{dd1} \approx 1.04$ V.

Assuming the load capacitances of a register and an adder are C_{reg} and C_{add}, respectively, we then obtain the relative energy consumption

$$\frac{E_1}{E_0} = \frac{(N/2)3(E_{reg} + E_{add})\big|_{V_{dd} = V_{dd1}}}{N(E_{reg} + E_{add})\big|_{V_{dd} = V_{dd0}}} = \frac{3}{2}\frac{(C_{reg} + C_{add})V_{dd1}^2}{(C_{reg} + C_{add})V_{dd0}^2} = \frac{3}{2}\left(\frac{V_{dd1}}{V_{dd0}}\right)^2.$$

Numerically $E_1/E_0 \approx 0.50$, and hence the energy savings become approx. 50%.