Application Specific Integrated Circuits for Digital Signal Processing Lecture 2

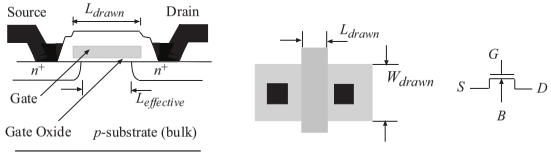
Oscar Gustafsson

Today's topics

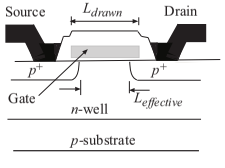
- Digital circuits
- Digital signal processing

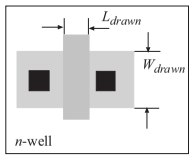
Digital circuits

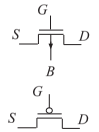
- Complementary metal-oxide semi-conductor (CMOS) technology commonly used and considered here
- MOS transistors (on p-substrate)
  - NMOS



  - PMOS







CMOS models

- Simple analog model

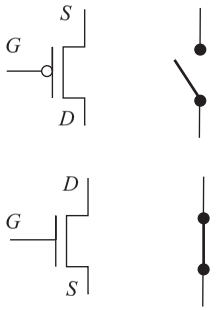
$$I_D = \begin{cases} 0 & V_{GS} - V_T < 0 \\ \beta\left(V_{GS} - V_T - \frac{V_{DS}}{2}\right) & V_{GS} - V_T > V_{DS} \\ \frac{\beta}{2}\left(V_{GS} - V_T\right)^2 & V_{GS} - V_T < V_{DS} \end{cases}$$

- V_T is the threshold voltage
- $\beta = \frac{\mu\epsilon W}{T_{ox} L}$
  - μ mobility (electrons or holes)
  - ϵ permitivity of SiO_2
  - T_{ox} gate oxide thickness
  - W, L width and length
- Difference between NMOS and PMOS – mobility
- Differently sized NMOS and PMOS transistors required for same current

CMOS models

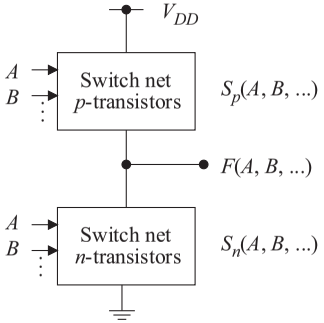
- For digital circuits $V_{DS} = V_{DD}$, $V_{GS} = 0$ or V_{DD}
- Current I_D is 0 or $\frac{\beta}{2}(V_{DD} - V_T)^2$
- Resulting structure can be seen as a switch

Transistor type	Gate input	Switch
PMOS	high (V_{DD})	off
	low (GND)	on
NMOS	high (V_{DD})	on
	low (GND)	off



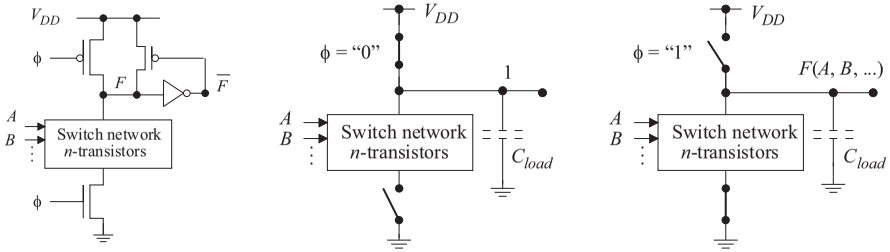
MOS logic

- Many different logic styles (ways to interconnect transistors)
- Static CMOS logic
  - Logic function $F(A, B, \dots)$ – Boolean algebra
  - Switching function $S(A, B, \dots)$ – Switching algebra
  - CMOS logic gate
    - P-transistors $S_P(A, B, \dots) = \overline{F(\overline{A}, \overline{B}, \dots)}$
    - N-transistors $S_N(A, B, \dots) = \overline{F(A, B, \dots)}$



MOS logic

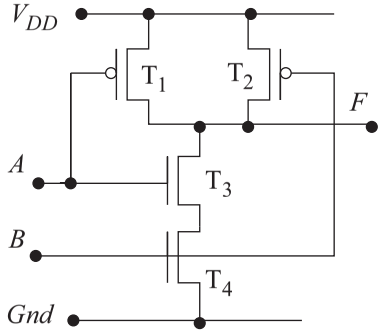
- Dynamic logic



- $\phi = 0$ – pre-charge
- $\phi = 1$ – evaluate

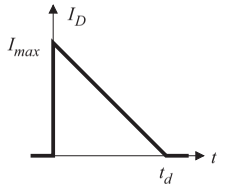
MOS logic example

- Two-input NAND-gate
  - $F(A, B) = \overline{AB}$
  - $S_N = \overline{F(A, B)} = AB$ – series connection
  - $S_P = F(\overline{A}, \overline{B}) = \overline{\overline{A}\,\overline{B}} = A + B$ – parallel connection



Propagation delay

- Discharge initial charge Q from load capacitance C_L through N-transistor $I_D = I_{\max}$ at $t = 0$, 0 at $t = t_d$



$$Q = C_L V_{DD} = \int_0^{t_d} I_D(t)dt \approx \frac{I_{\max} t_d}{2} = \frac{\beta}{2}(V_{GS} - V_T)^2 \frac{t_d}{2} \quad (1)$$

- Discharge time

$$t_d = \frac{4 C_L V_{DD}}{\beta (V_{DD} - V_T)^2} \quad (2)$$

Propagation delay

- For deep-submicron processes ($\alpha \approx 1.2$ to 1.55)

$$t_d = \frac{4 C_L V_{DD}}{\beta (V_{DD} - V_T)^\alpha} \quad (3)$$

  due to velocity saturation
- Propagation delay: $\tau = -t_d \ln(0.5) \approx 0.69 t_d$

Power dissipation

- Important for:
  - Battery life time
  - Cooling
  - Avoiding metal migration
- Energy consumed by charging to V_{DD} and discharging a capacitance C

$$E = C V_{DD}^2 \quad (4)$$

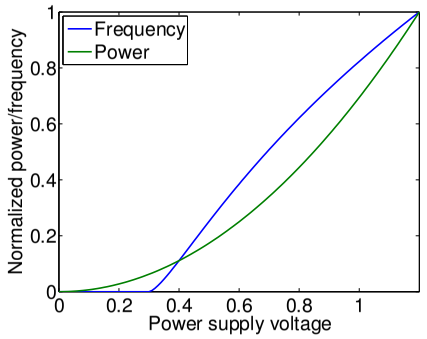
- Power consumption

$$P = a f_{clk} C V_{DD}^2 \quad (5)$$

  - a – switching activity
  - f_{clk} – clock frequency

Power supply voltage scaling

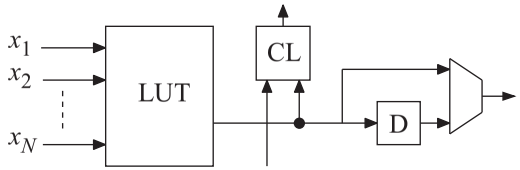
- Normalized maximum frequency and power as function of power supply voltage



- Power reduces faster compared to frequency
- Often beneficial to design a too fast circuit and lower the power supply voltage

Field-Programmable Gate-Arrays (FPGA)

- The basic building block of an FPGA is a look-up table (LUT) storing the truth table of the logic function
- Typical LUT size (N) between four and six



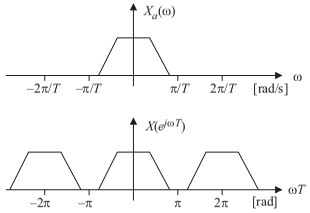
- Optional register at output
- Common to have carry logic to speed up addition

FPGA

- Larger FPGAs have dedicated multipliers or even more complicated arithmetic blocks including adder/accumulator and bit-operations
- Typical sizes are 18×18 or 18×25 bits with 48 bits accumulator
- Dedicated large memories are also commonly available

Digital Signal Processing

- Analog signal spectrum $X_a(\omega)$
- Sampled signal spectrum $X\left(e^{j\omega T}\right)$, sample period T



- Fourier transform

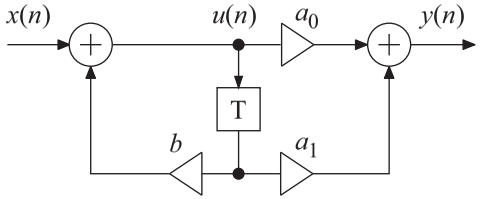
$$X\left(e^{j\omega T}\right) = \sum_{n=-\infty}^{\infty} x(nT)e^{-jn\omega T} \tag{6}$$

- Z-transform ($z = e^{j\omega T}$)

$$X(z) = \sum_{n=-\infty}^{\infty} x(nT)z^{-n} \tag{7}$$

Signal flow graph (block diagram)

- Time-domain representation of algorithm



$$u(n) = x(n) + bu(n-1) \tag{8}$$

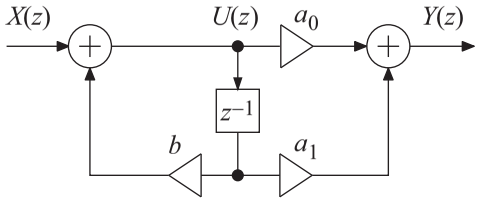
$$y(n) = a_0 u(n) + a_1 u(n-1) \tag{9}$$

$$y(n) = by(n-1) + a_0 x(n) + a_1 x(n-1) \tag{10}$$

- T – sample delay (algorithmic), D – flip-flop/register

Signal flow graph (block diagram)

- Z-domain (frequency-domain) representation of algorithms



$$U(z) = X(z) + bz^{-1}U(z) \Rightarrow U(z) = \frac{X(z)}{1 - bz^{-1}} \tag{11}$$

$$Y(z) = a_0 U(z) + a_1 z^{-1} U(z) = \frac{X(z)\left(a_0 + a_1 z^{-1}\right)}{1 - bz^{-1}} \tag{12}$$

$$H(z) = \frac{Y(z)}{X(z)} = \frac{a_0 + a_1 z^{-1}}{1 - bz^{-1}} \tag{13}$$

Discrete Fourirer Transform (DFT)

- The N-point DFT is the Fourier transform of N samples evaluated at N equally spaced points on the unit circle $\omega T = \frac{2\pi k}{N}$

$$X(k) = X\left(e^{j\omega T}\right)\Big|_{\omega T = \frac{2\pi k}{N}} = \sum_{n=0}^{N-1} x(n) e^{\frac{-j2\pi nk}{N}} = \sum_{n=0}^{N-1} x(n) W_N^{nk} \tag{14}$$

- Twiddle factor: $W_N = e^{\frac{-j2\pi}{N}}$
- Inverse DFT (IDFT)

$$x(n) = \frac{1}{N}\sum_{n=0}^{N-1} X(k) e^{\frac{j2\pi nk}{N}} = \frac{1}{N}\sum_{n=0}^{N-1} X(k) W_N^{-nk} \tag{15}$$

- Used in spectral analysis, OFDM, RADAR, and more
- Direct computation requires N^2 complex multiplications

Fast Fourier Transform (FFT)

- Assume that the N is a non-prime number such that $N = PQ$
- The DFT equation can be rewritten as P Q-point DFTs, N twiddle factor multiplications, and Q P-point DFTs

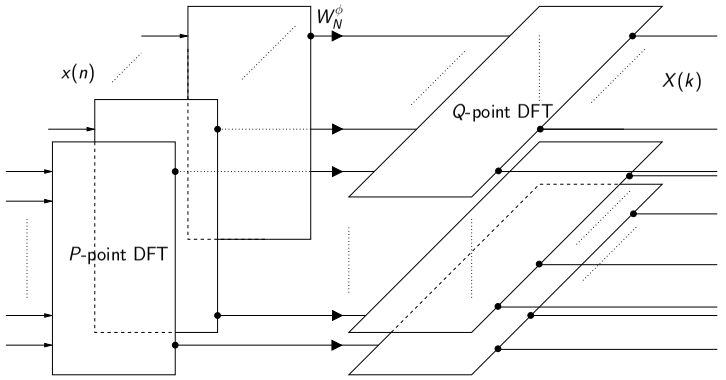
$$X\left[k_1 + Pk_2\right] = \underbrace{\sum_{n_2=0}^{Q-1}\left[\overbrace{\left(\sum_{n_1=0}^{P-1} x\left[Qn_1 + n_2\right] W_P^{n_1 k_1}\right)}^{P\text{-point DFT}} \overbrace{W_N^{t_2 n_1}}^{\text{TF mult}}\right] W_Q^{t_2 n_2}}_{Q\text{-point DFT}} \tag{16}$$

where $0 \leq k_1 \leq P-1$ and $0 \leq k_2 \leq Q-1$

- Complexity $P^2Q + Q^2P + N = N\underbrace{(P+Q+1)}_{<N=PQ} < N^2$

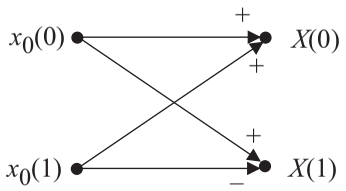
FFT

- Block diagram



FFT

- If P and/or Q are non-prime number, they can be split further
- The most common case is $N = 2^M$
- Can be split down to 2-point DFTs
- 2-point DFT (butterfly operation)



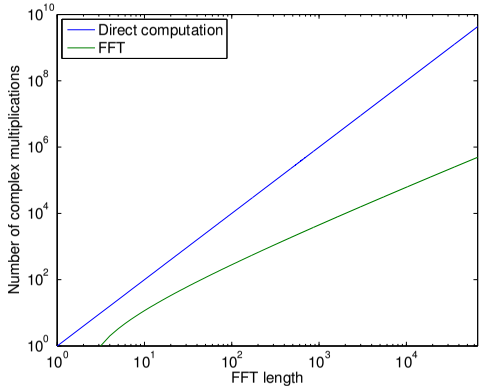
- Complexity: $\frac{N}{2}\log_2 N = \frac{NM}{2}$ butterflies, $N(\log_2 N - 1) = N(M-1)$ complex multipliers
- Many (roughly half, depending on in which order the smaller DFTs are derived) of the complex multipliers will have coefficients $= 0$

FFT algorithms

- Standard radix-2 algorithms
  - Radix-2 decimation-in-time (DIT) FFT (Cooley-Tukey) – split as 2 and $N/2$, then the $N/2$ as 2 and $N/4$ and so on
  - Radix-2 decimation-in-frequency (DIF) FFT (Cooley-Tukey) – split as $N/2$ and 2, then the $N/2$ as $N/4$ and 2 and so on
  - Radix-2^2 DIT/DIF – split as 4 and $N/4$, then split the 4 as 2 and 2, $N/4$ as 4 and $N/16$ and so on
    - The TF multiplier between the 2 and 2 terms in the 2^2 will be a W_4 TF multiplier, so only multiplication with 1 and $-j$
  - Can be generalized as 2^i with different resolutions of the different TF multipliers
- Can of course have a larger DFT as the smallest size, radix-4 etc.

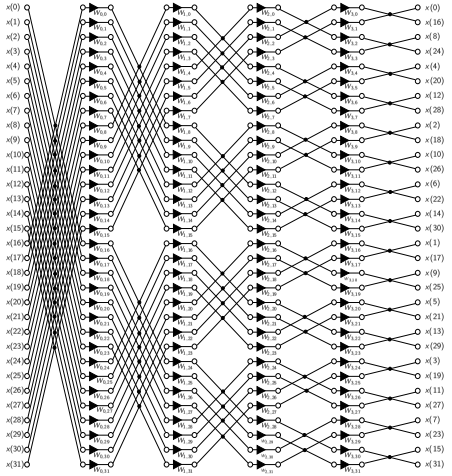
FFT complexity

- Rough complexity estimation for radix-2 algorithms



- Note that the FFT is only an efficient (class of) algorithm(s) to compute the DFT
- Often the terms are used interchangably

32-point FFT SFG



- Can be used for any radix-2 algorithm (include 2^i), the difference is the twiddle factor values
- The outputs are provided in bit-reversed order, i.e., the output at row $6_{10} = 00110_2$ corresponds to value $01100_2 = 20_{10}$

Case Study I – FFT

- In the first case study we will design an FFT processor with the following specifications (note the last-millenium specs...):
  - 1024 points
  - 2000 FFTs/s $\Rightarrow$ 0.5 ms per FFT
  - Connected to a computer via a 32-bit bus clocked at 16 MHz
  - 16 + 16 bits real and complex data input and output

Case Study I – FFT

- Number of butterfly operations per FFT: $\frac{N}{2}\log_2(N) = 5120$
- Assume that the processor is idle during data transfer (otherwise more memory would be required)
- Time to transfer data: $\frac{1024\times 2}{16\times 10^6} = 0.128$ ms
- Remaining time for FFT: $0.5 - 0.128 = 0.372$ ms
- Assume processing element (PE) with bit-serial butterfly and complex multiplier
  - 24 clock cycles required per operation
  - Max 220 MHz clock frequency
- Number of PEs

$$N_{PE} = \frac{N_{clk}N_{OP}}{t_{FFT}f_{\max}} = \frac{24\times 5120}{0.372\times 10^{-3}\times 220\times 10^6} \approx 1.5 \quad (17)$$

- Use two PEs

Case Study I – FFT

- Two PEs lead to a minimum clock frequency of 165.2 MHz
- For each PE operation we need to read and write two complex values
- The memory access rate is

$$f_{mem} = \frac{(2+2)5120}{0.372\times 10^{-3}} \approx 55\times 10^6 \text{complex values/s} \quad (18)$$

- Choose to use two memories to avoid multiple accesses per operation to a single memory
- Also, assume the memory rate is a multiple of the I/O rate $\Rightarrow$ 32 MHz
- The new time for an FFT is now 0.32 ms and the clock frequency 192 MHz
- Finally, with these parameters the resulting throughput is 2232 FFTs/s

Discrete cosine transform (DCT)

- DCTs are used in e.g. image coding
- Several different DCT transforms proposed
- For JPEG, the following 2-D DCT is used

$$G(u,v) = \sum_{x=0}^{7}\sum_{y=0}^{7}\frac{g(x,y)}{\alpha(u,v)}\cos\left[\frac{\pi}{8}\left(x+\frac{1}{2}\right)u\right]\cos\left[\frac{\pi}{8}\left(y+\frac{1}{2}\right)v\right] \quad (19)$$

  where $\alpha(u,v)$ is a scaling factor
- Can be separated into 1-D DCTs that operate on first rows and then columns
- Fast structures available, but not as regular as for the DFT