CSE4210 Architecture & Hardware for DSP

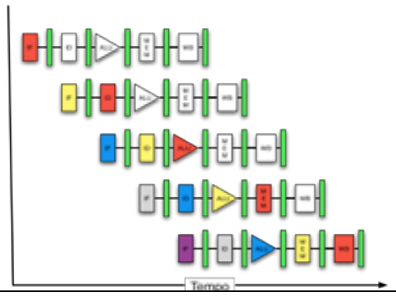
Chapter 6

Pipelining and Parallel Processing

Instructor: Prof. Peter Lian
Department of Electrical Engineering & Computer Science
Lassonde School of Engineering
York University

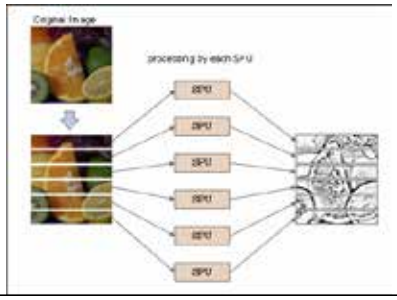
Introduction

- Pipelining
  - Pipelining is a set of data processing elements connected in series, where the output of one stage is the input of the next one.
  - Leads to a reduction in the critical path
  - Either increases the clock speed (or sampling speed) or reduces the power consumption at same speed in a DSP system.

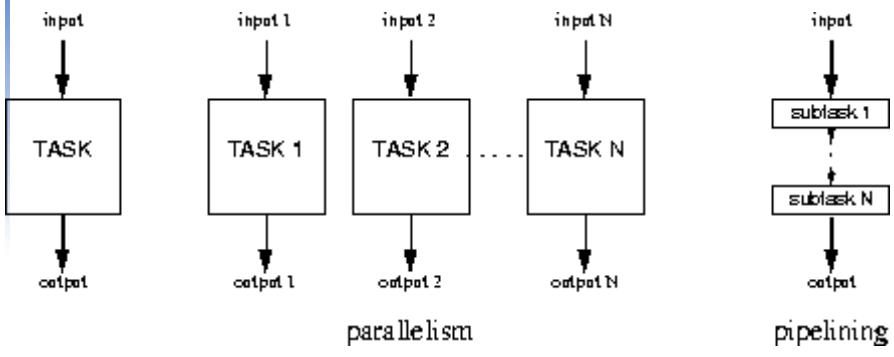


Introduction

- Parallel processing
  - Multiple outputs are computed in parallel in a clock period.
  - The effective sampling speed is increase by the level of parallelism.
  - Can also be used to reduce the power consumption.



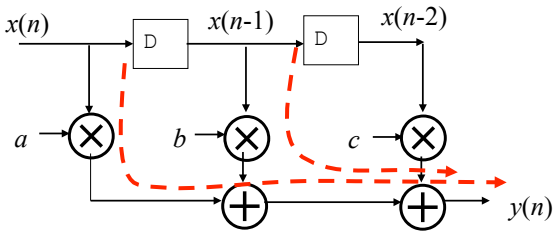
Pipelining vs Parallel



Pipelining

Example of a 3-tap FIR Filter

$$y(n) = a * x(n) + b * x(n-1) + c * x(n-2)$$



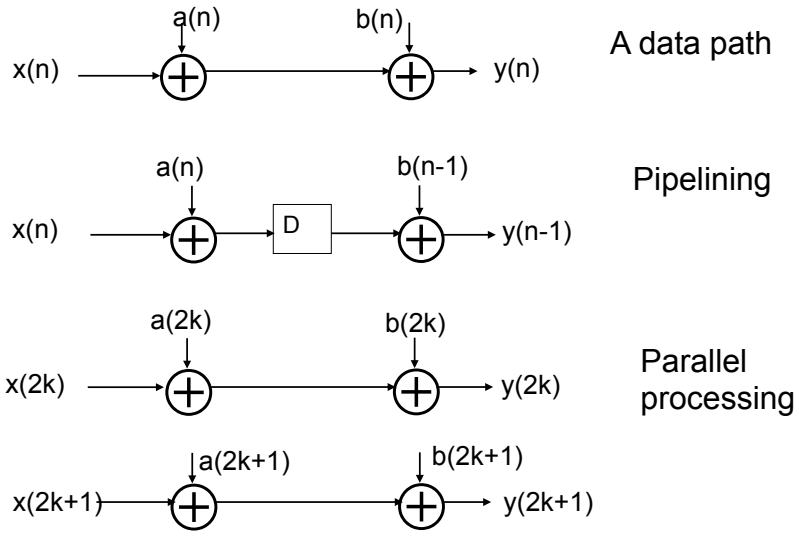
The critical path here is two additions and one multiplication

$$T_s = 2T_{add} + T_{mul} \quad , \quad f_s = 1/(2T_{add} + T_{mul})$$

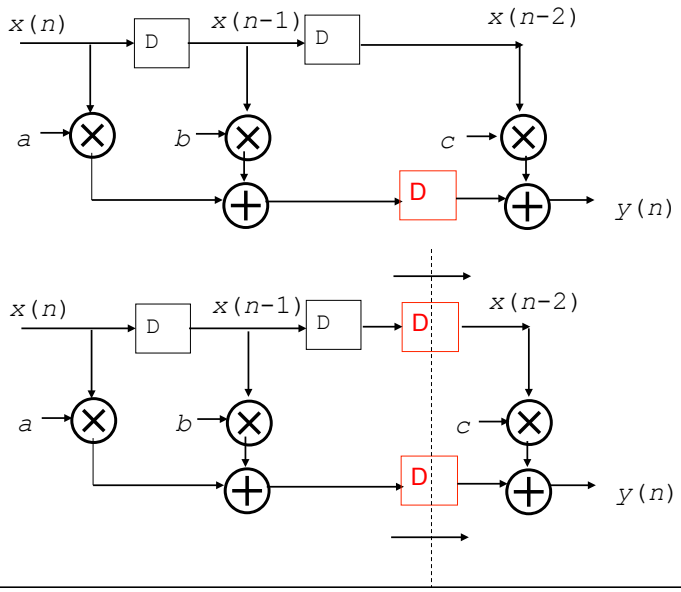
How to increase f_s ?

- What if the application requires a faster input rate?
- Pipelining: reduce the effective critical path by introducing pipelining latches along the critical data path.
- Parallel processing: increases the sampling rate by replicating hardware so that several inputs can be processed in parallel and several outputs can be produced at the same time

2-Level Pipelining and Parallel

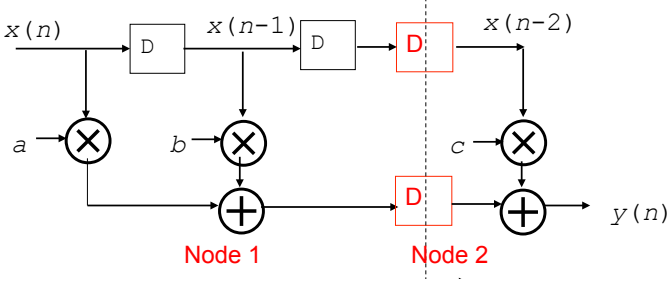


Pipelining



Pipelined Implementation

- By introducing 2 additional latches in the filter, the critical path is reduced from $2T_{add}+T_{mul}$ to $T_{add}+T_{mul}$.



Clock	Input	Node 1	Node 2	Node 3	Output
0	x(0)	ax(0)+bx(-1)	—	—	—
1	x(1)	ax(1)+bx(0)	ax(0)+bx(-1)	cx(-2)	y(0)
2	x(2)	ax(2)+bx(1)	ax(1)+bx(0)	cx(-1)	y(1)
3	x(3)	ax(3)+bx(2)	ax(2)+bx(1)	cx(0)	y(2)

Pipelining Pros and Cons

- Advantages
  - Could be used to reduce power and/or to increase clock rate (speed)
- Disadvantages
  - Increases number of delay elements (latches or flip-flops)
  - Increases latency (the difference in the availability of the first output data in the pipelined system and the sequential system)

Pipelining Latches

- The speed of an architecture is limited by the longest path between any 2 latches or between an input and a latch or between a latch and an output or between the input and output
- This longest path or the “critical path” can be reduced by suitably placing the pipelining latches in the architecture.
- The pipelining latches can only be placed across any feed-forward cutset of the graph.

Definition of Cutset

- **Cutset**: is a set of edges of a graph if removed, the graph becomes disjoint.
- **Feed forward cutset**: a cutset where the data moves in the forward direction on all the edges in the cutset.
- We can place latches on a **feed-forward cutset** without affecting the functionality of the graph.

Activity 1

In the signal-flow graph (SFG) in Fig. a, the computation time for each node is assumed to be 1 u.t.

(1) Calculate the critical path computation time.

(2) The critical path has been reduced to 2 u.t. by inserting 3 extra latches as shown in Fig. b. Is this a valid pipelining? If not, how to get a valid one.

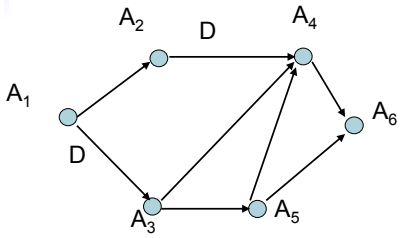


Fig. a

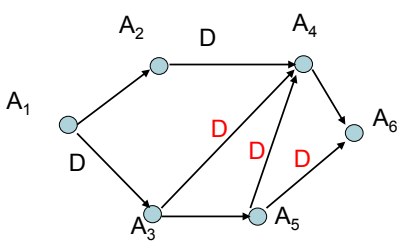
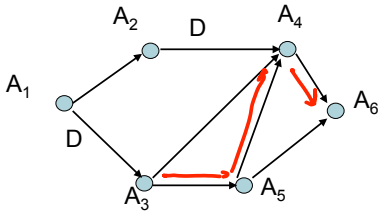


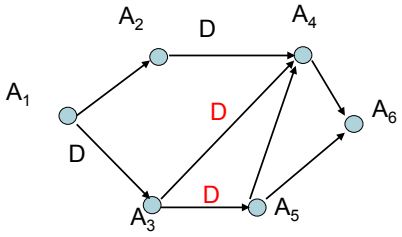
Fig. b

Activity 1

Critical Path =4 u.t.



Critical Path =3 u.t.



Critical Path =2 u.t.

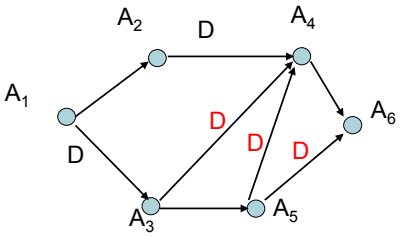


Fig. b

Activity 1

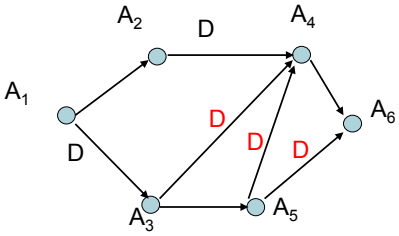
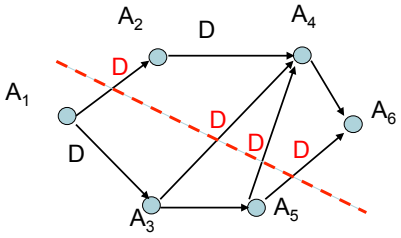


Fig. b

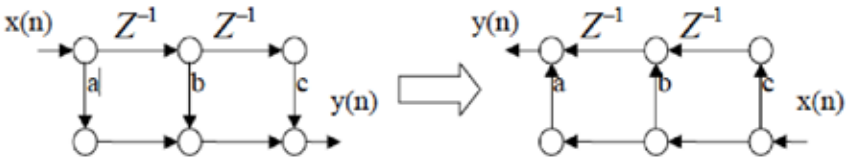
Not a valid pipelining implementation



A valid pipelining implementation

Transposed FIR Filter SFG

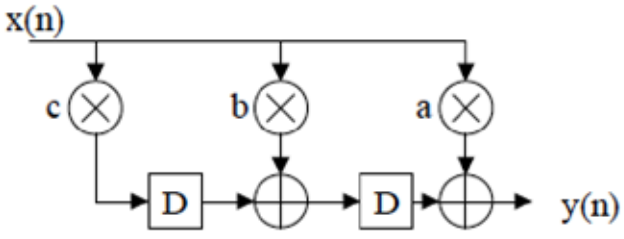
- Reversing the direction of all the edges in a given SFG, and interchanging the ,input and output preserves the functionality of the system.



$T_M + 2T_a$

$T_M + T_a$

Data Broadcast Structure



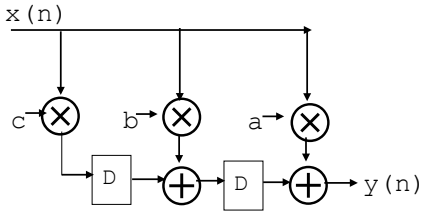
$T_M + T_a$

Fine-Grain Pipelining

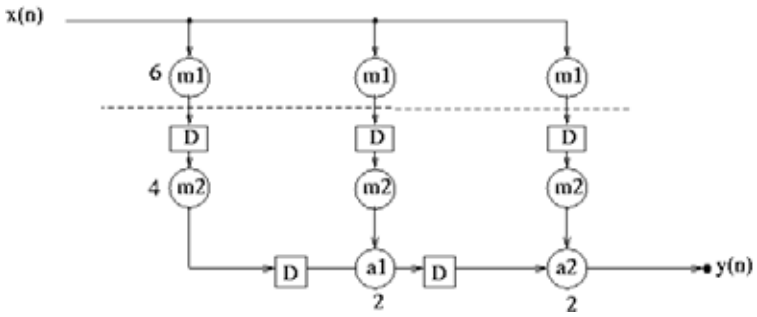
Multiplication time = 10

Addition time = 2

Critical path = ?



Fine Grain Pipelining



Parallel Processing

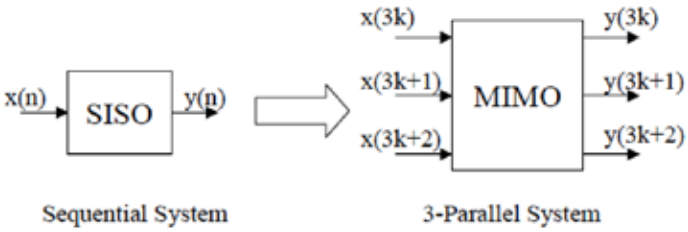
Parallel Processing

- Parallel processing and pipelining techniques are duals each other: if a computation can be pipelined, it can also be processed in parallel. Both of them exploit concurrency available in the computation in different ways.
- Example for parallel FIR system
  - A single-input single-output (SISO) FIR filter can be converted into an multiple-input multiple-output system (MIMO) in order to obtain a parallel processing structure

Parallel Processing Example

$$y(n) = ax(n) + bx(n-1) + cx(n-2)$$

$$y(3k) = ax(3k) + bx(3k-1) + cx(3k-2)$$
$$y(3k+1) = ax(3k+1) + bx(3k) + cx(3k-1)$$
$$y(3k+2) = ax(3k+2) + bx(3k+1) + cx(3k)$$



Sequential System

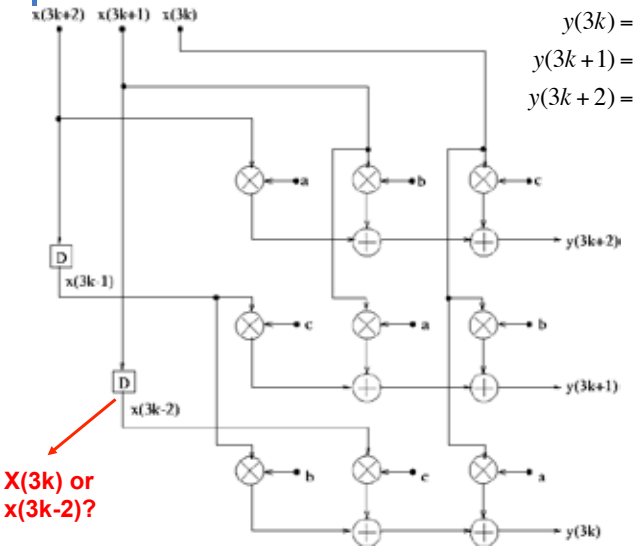
3-Parallel System

Parallel Processing

- Parallel processing system is also called block processing, and the number of inputs processed in a clock cycle is referred to as the block size.
- In the example, at the k-th clock cycle, 3 inputs x(3k), x(3k+1), and x(3k+2) are processed and 3 samples y(3k), y(3k+1) and y(3k+2) are generated at the output.

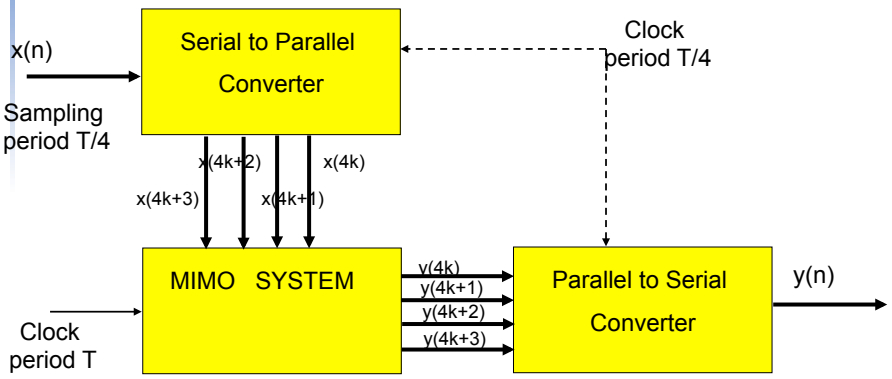
Parallel Processing of a Filter

$$y(3k) = ax(3k) + bx(3k-1) + cx(3k-2)$$
$$y(3k+1) = ax(3k+1) + bx(3k) + cx(3k-1)$$
$$y(3k+2) = ax(3k+2) + bx(3k+1) + cx(3k)$$



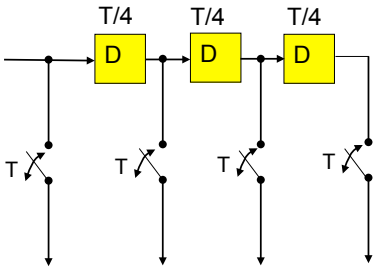
X(3k) or x(3k-2)?

Complete Parallel System

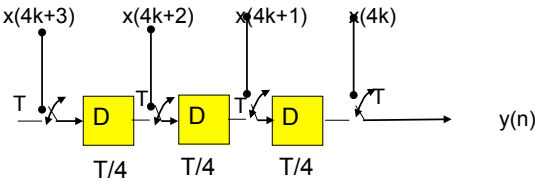


S/P and P/S Converter

Serial-to-parallel converter

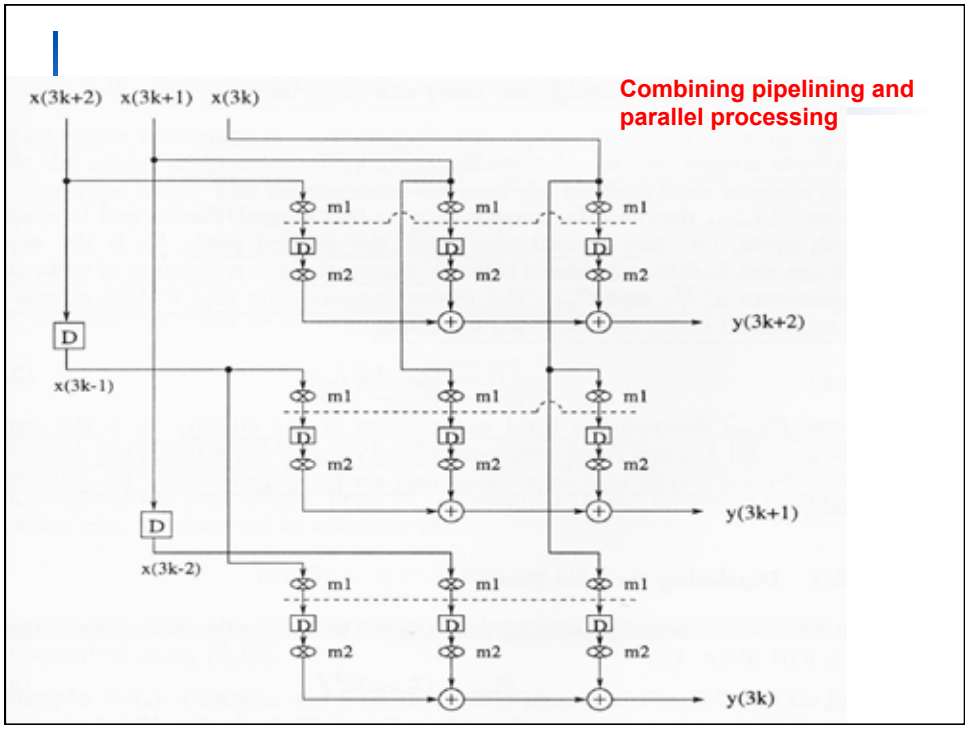


Parallel-to-serial converter



Parallel Processing

- Why use parallel processing? It increases the hardware.
- There is a limit for the use of pipelining, you may not be able to pipeline a functional unit beyond a certain limit
- Also, I/O usually imposes a bound on the cycle time (communication bound)



CSE4210 Architecture & Hardware for DSP

Low Power Design Exploring Pipelining and Parallel Processing

Low Power

- Two important formulas:
  - Power consumption & propagation delay

$$P = C_{total} V_o^2 f$$

$$T_{pd} = \frac{C_{\text{charge}} V_o}{k(V_o - V_t)^2}$$

C_{total} is the total capacitance of the circuit, V_o is the supply voltage. C_{charge} is the capacitance to be charged/ discharged in a single clock cycle.

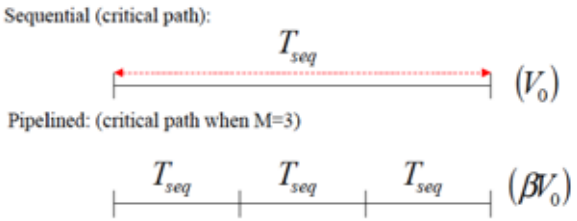
- Pipelining and parallel processing could be used to minimize power or execution time.

Pipelining for Low Power

- The power consumption in the original sequential FIR filter

$$P_{seq} = C_{total} V_o^2 f, \; f = \frac{1}{T_{seq}}$$

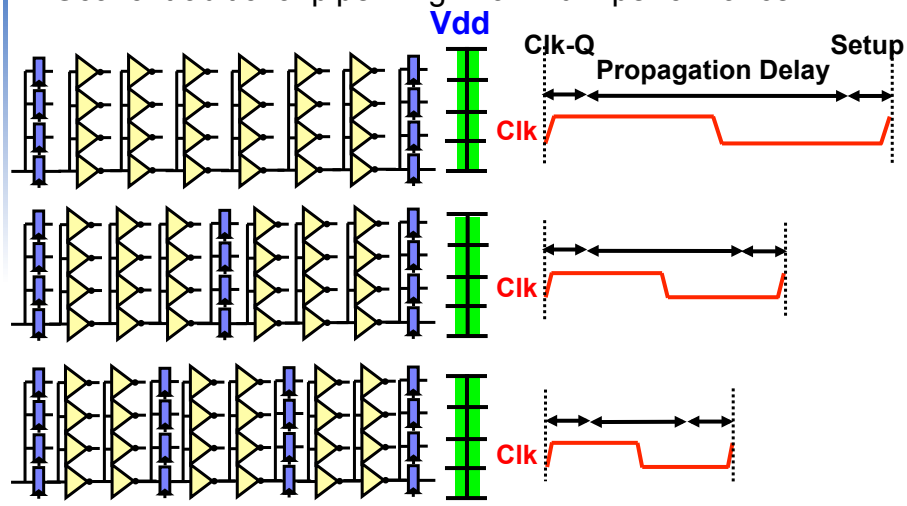
- In *M*-level pipelining FIR filter, critical path is reduced to 1/*M* of its original length.



- If the same clock speed is maintained, then pipelined circuit can be “slow”.

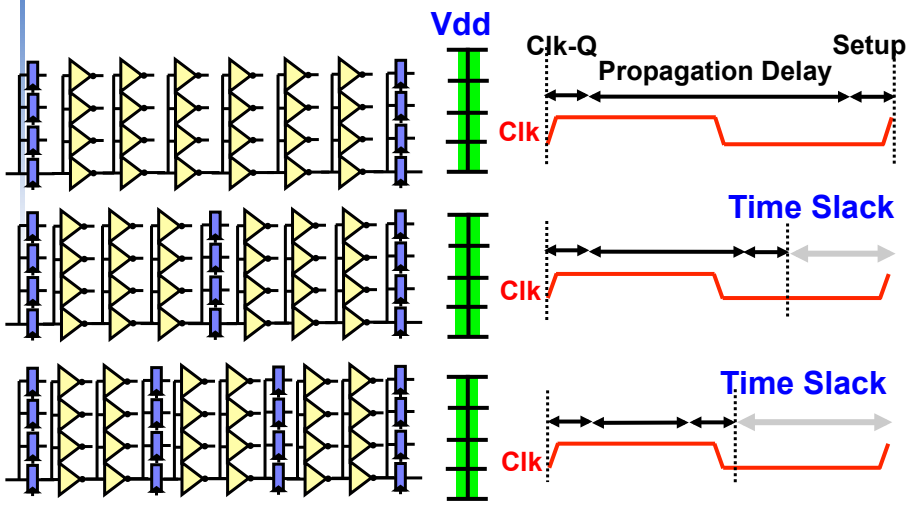
Pipelining for Low Power

- Goal of traditional pipelining: Maximum performance

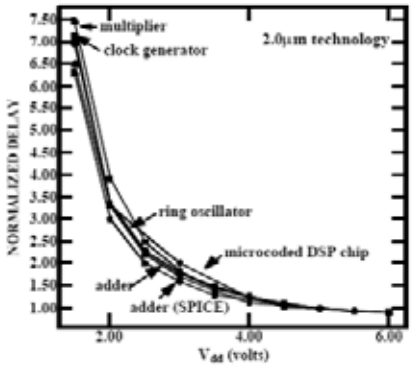


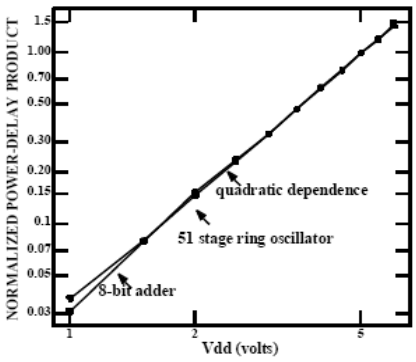
Pipelining for Low Power

- Goal: Low-Power, Fixed Throughput



Slow Circuit → Low Voltage





- Lowering Vdd increases delay
- Lowering Vdd improves energy consumption

Pipelining for Low Power

- In *M*-level pipelined filter running at the same clock rate as non-pipelined design, the power consumption is given by:

$$P_{pip} = C_{total}\left(\beta V_0\right)^2 f = \beta^2 C_{total} V_o^2 f = \beta^2 P_{seq},\ 0 < \beta < 1$$

- How to find β?

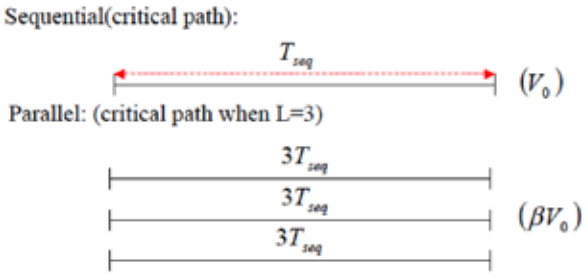
$$T_{seq} = \frac{C_{\text{charge}} V_o}{k(V_o - V_t)^2}, T_{pip} = \frac{\frac{C_{\text{charge}}}{M}\beta V_o}{k(\beta V_o - V_t)^2}$$

$$T_{seq} = T_{pip} \rightarrow \frac{C_{\text{charge}} V_o}{k(V_o - V_t)^2} = \frac{C_{\text{charge}} \beta V_o}{kM(\beta V_o - V_t)^2}$$

$$\rightarrow M\left(\beta V_o - V_t\right)^2 = \beta(V_o - V_t)^2$$

Parallel processing

- What happen for L parallel System?
- Total capacitance increased by L
- Same performance, increase T by L
- More time to charge C_{charge} → decrease V



Parallel processing

- The propagation delay of the L-parallel system is

$$LT_{seq} = L\frac{C_{\text{charge}}V_o}{k(V_o - V_t)^2}, \quad T_{par} = \frac{C_{charge}\,\beta V_o}{k(\beta V_o - V_t)^2}$$

$$LT_{seq} = T_{par} \rightarrow L\frac{C_{\text{charge}}V_o}{k(V_o - V_t)^2} = \frac{C_{charge}\,\beta V_o}{k(\beta V_o - V_t)^2}$$

$$\rightarrow L(\beta V_o - V_t)^2 = \beta(V_o - V_t)^2$$

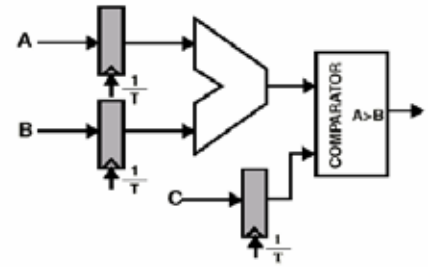
- Power of parallel system is

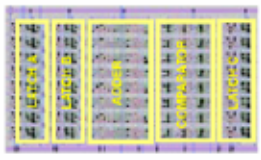
$$P_{par} = (LC_{total})(\beta V_0)^2 \frac{f}{L} = \beta^2 P_{seq}, \; 0 < \beta < 1$$

Examples

- Example of pipelined system
  - Please read textbook for Example 3.4.1 on page 75.
- Example of parallel system
  - Please read the Example 3.4.2 in the textbook on page 77.

Parallel for Low Power



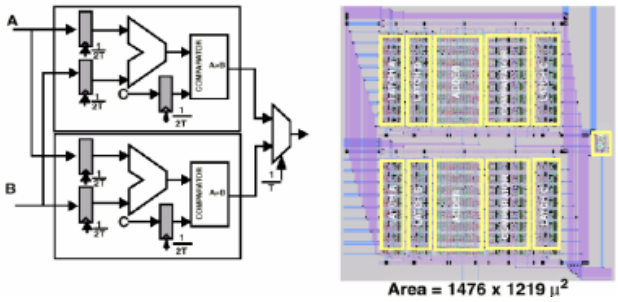


Area = 636 x 833 μ^2

- Critical path delay:
  T_{adder} +$T_{comparator}$=25ns->f_{ref}=40MHz
- Total capacitance being switched = C_{ref}
- V_{dd}=V_{ref}=5V
- Power for reference datapath: P_{ref}=$C_{ref}V_{ref}^2f_{ref}$

*Chandrakasan, IEEE JSSC' 92

Parallel as a Low Power Tool



- The clock rate can be reduced by half with the same throughput -> $f_{par}=f_{ref}/2$
- $V_{par}=V_{ref}/1.7$, $C_{par}=2.15C_{ref}$
- $P_{par}=(2.15C_{ref})(V_{ref}/1.7)^2(f_{ref}/2)=0.36P_{ref}$

Parallel as a Low Power Tool

Architecture type	Voltage	Area	Power
Simple datapath (no pipelining or parallelism)	5V	1	1
Pipelined datapath	2.9V	1.3	0.39
Parallel datapath	2.9V	3.4	0.36
Pipeline-Parallel	2.0V	3.7	0.2