VLSI Design
Lecture 16: Clocking disciplines

Shaahin Hessabi
Department of Computer Engineering
Sharif University of Technology
Adapted, with modifications, from lecture notes prepared by the book’s author (from Prentice Hall PTR)
Flip-flop-based sequential machines
Flip-flop rules

- Inputs change after clock (ϕ) edge.
- Inputs must stabilize before next clock edge.
- Rules allow changes to propagate through combinational logic for next cycle.
- Flip-flop outputs hold current-state values for next-state computation.

![Diagram showing positive clock edge and flip-flop states: stable and changing.]
Latch-based machines

- Latches do not cut combinational logic when clock is active.
- Latch-based machines must use multiple ranks of latches.
- Multiple ranks require multiple phases of clock.
Two-sided latch constraint

- Latch must be open less than the shortest combinational delay.
- Period between latching operations must be longer than the longest combinational delay.
- Note: difference between shortest and longest combinational delay may be large ($\text{sum}_0$ vs. $\text{sum}_{31}$).
Latch shoot-through

Latch may allow data to shoot through:

![Diagram of a latch circuit]
Strict two-phase clocking discipline

- Strict two-phase discipline is conservative but works.
- Can be relaxed later with proper knowledge of constraints.
- Strict two-phase machine makes latch-based machine behave more like flip-flop design, but requires multiple phases.
Strict two-phase architecture

Phases must not overlap:

non-overlap region
Why it works

- Each phase has a **one-sided** constraint: phase must be long enough for all combinational delays.
- If there are no combinational loops, phases can always be stretched to make that section of the machine work.
- Total clock period depends on sum of phase periods.
Clocking types

- Logic on different phases operate at different times: can’t mix signals from different phases.
- Primary inputs must obey the same rules as internal signals.
- Clocking types are bookkeeping that help us ensure that machine structure is valid.
Stable signals

- A logic signal is always stable during one phase: phase in which the latch which produced it is not active.
- Easiest to think of machine behavior in terms of stable signals, though signals propagate while not stable.
Clocks are separate types: $\phi_1$, $\phi_2$.

Two types of stable data signal:
- stable $\phi_1$ ($s \phi_1$)
- stable $\phi_2$ ($s \phi_2$)

A stable signal has a complementary valid signal:
- stable $\phi_2$ ($s \phi_2$) = valid $\phi_1$ ($v \phi_1$)
Stable data signal

- \( \phi_1 \) is an inactive clock.
- \( \phi_2 \) becomes stable \( \phi_2 \) becomes valid at end of \( \phi_1 \).
- Stable until latch feeding this logic goes active.

Modern VLSI Design 4e: Chapter 5
Sharif University of Technology
Slide 13 of 25
How clocking types combine

\[ S\phi_1 \quad S\phi_1 \quad S\phi_1 \quad \phi_1 \quad S\phi_1 \quad S\phi_2 \]
Clocking types in the two-phase machine

- Combinational logic does not change type of signal.
- Primary inputs must be compatible.
- Latches change signals from one clock type to another.
- In strict system, never mix clocks with data signals in combinational logic.
Two-coloring
Example: shift register

- Want to displace bit by $n$ registers in $n$ cycles.
- Each register requires two phases:
Shift register layout

Forms a linear array:
Shift register operation

\[ \phi_1 = 1, \quad \phi_2 = 0 \]

\[ \phi_1 = 0, \quad \phi_2 = 1 \]
Non-strict disciplines

- Some relaxation of the rules can be useful:
  - reduce area;
  - increase performance.

- Rules must be relaxed in a way that ensures the machine will still work.
Qualified clocks

- Use logic to generate a clock signal which is not always active.
- Qualification must not introduce glitches into the clock: glitches violate the fundamental definition of a clock by introducing extra edges.
- Use stable signals to qualify clocks.
Uses of qualified clocks

- May want to conditionally load a register.
- May qualify a clock to turn off machine for low-power operation.
- Latch must keep its value during inactive period.
- Difficult to ensure that logic value will come high in time: use quasi-static latch.
Recirculating latch

Synchronous design ensures that the system works, in spite of relaxation of the rules.
Qualified clocks and skew

- Logic in the clocking path introduces delay.
- Delay can cause clock to arrive at latches at different times, violating clocking assumptions.
- When designing qualification logic:
  - minimize and check skew;
  - sharpen clock edge.
Qualification skew example

![Diagram of qualification skew example](image-url)