PVT-aware Self-tuning Circuits

Presented by: Navid Toosizadeh

Supervised by: Prof. Safwat G. Zaky

Process, Voltage, Temperature (PVT) Variations

- Process variations make the quality of fabricated chips less predictable.
- How to deal with it?
  - Use more & more conservative delay estimations (assume worst-case PVT)
- Resulting products?
  - Large and leaky circuits (sub-100nm)

Process, Voltage, Temperature Variations

Undesired Over-engineering?

Is there any better way to design?

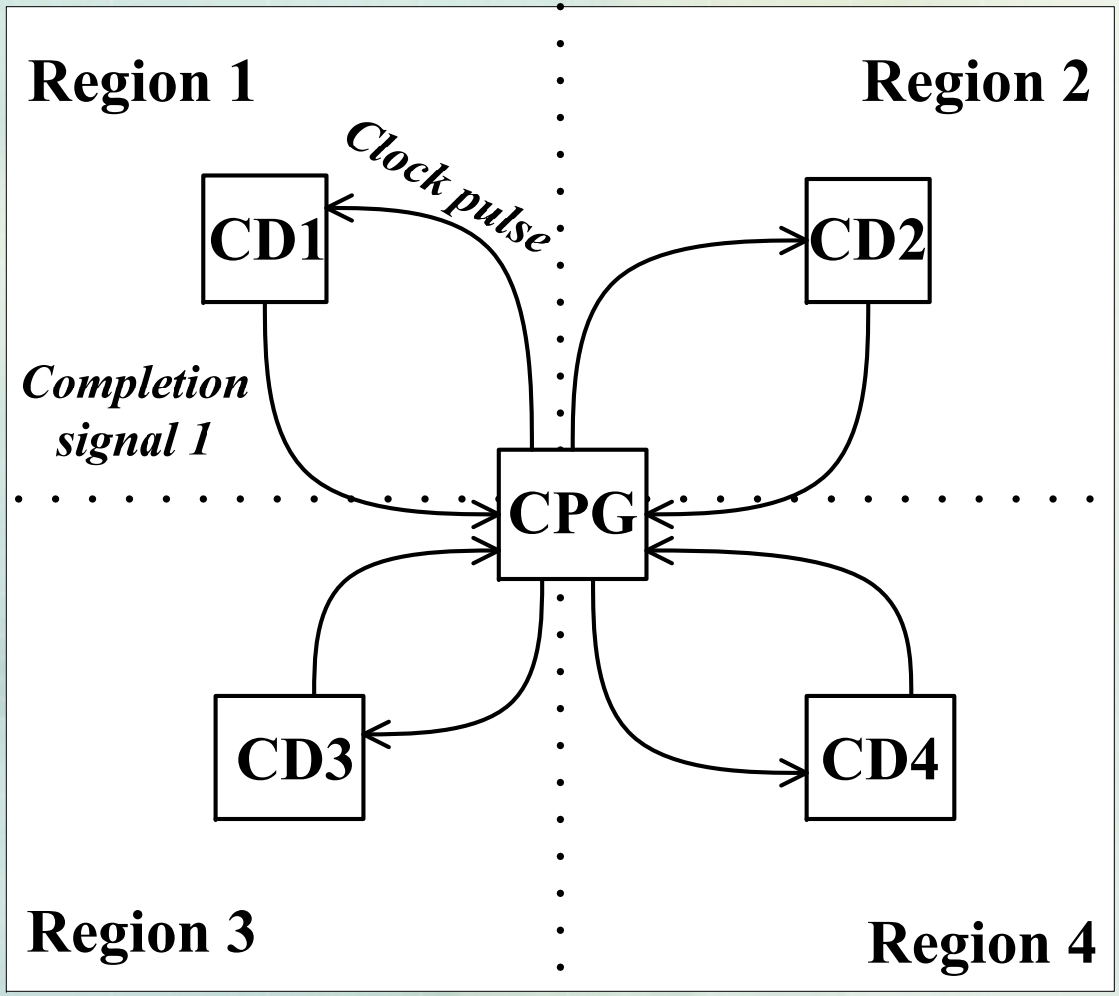
Self-tuning Circuits

- They produce the best-possible results, given the ***PRESENT*** process, voltage, temperature.
- Don't worry about variations, design for **TYPICAL** conditions (**not worst case**)
- What if the chip is actually exposed to the worst conditions?
  - An **on-chip** circuitry **auto-corrects** for a poor performance due to variations

Self-tuning Mechanism

CD: *Composed of delay elements that match the critical path delay.*

CPG: *Creates a new clock pulse when all regions sent their completion signal.*



Post-layout Results: Typical corner

- Two DLX microprocessors, implemented in **90nm**: one traditional (fixed-clock) and one PVT-aware.
- Results under **TYPICAL** PVT conditions (temp=25º C):

Design	Av. Leakage (mW)	Av. Total power (mW)	Total Ex. Time (mS)
Traditional	1.242	38.402	6.492
PVT-aware	0.125	30.223	6.492
Change	**10X Reduction**	**21% Reduction**	**Similar**

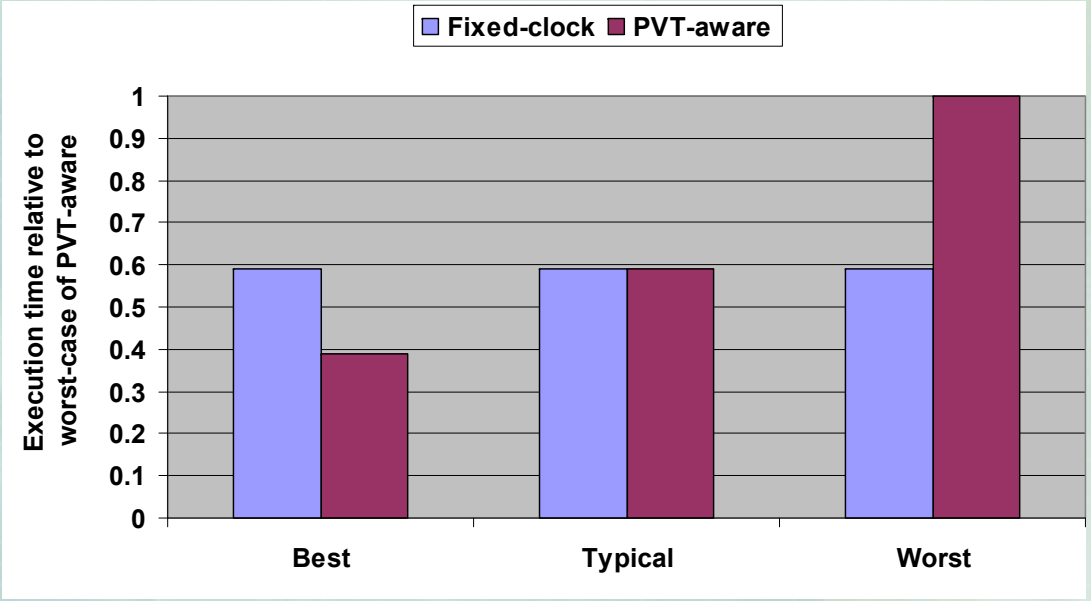
Post-layout Results: Worst corner

- Results under **WORST** PVT conditions (temp=125º C):

Design	Av. Leakage (mW)	Av. Total power (mW)	Total Ex. Time (mS)	**Power× delay (µ J)**
Trad.	**59.193**	87.216	6.502	**567.08**
PVT-aware	**8.092**	22.344	11.023	**246.30**

Resilience to PVT variations, Amenability to voltage scaling

- Simulations verified that the PVT-aware processor produces the best-possible results with the inter- and intra-chip variations.
- The processor is amenable to voltage scaling.

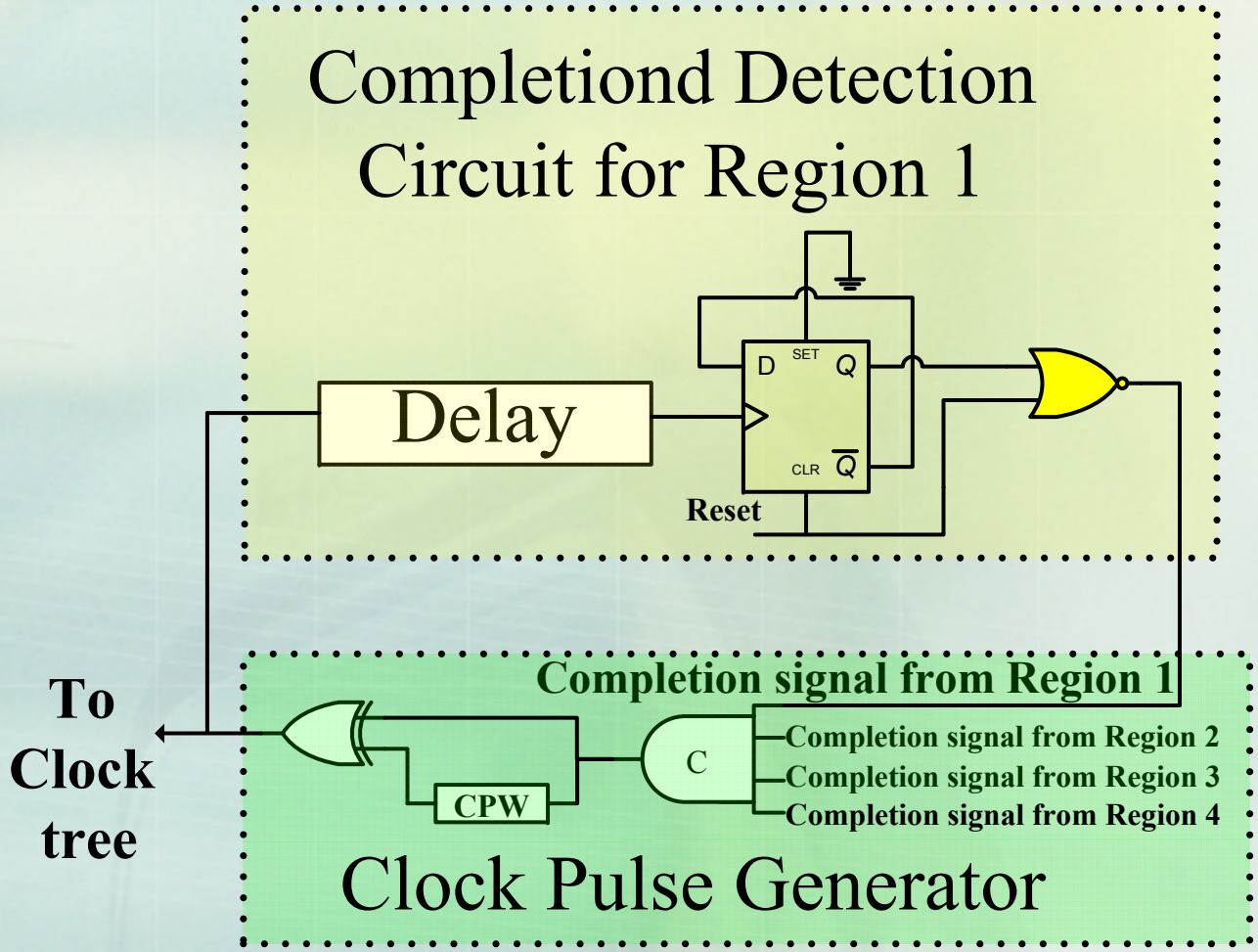


Conclusion

PVT-aware self-tuning circuits are viable solutions for today's shrinking technologies to:

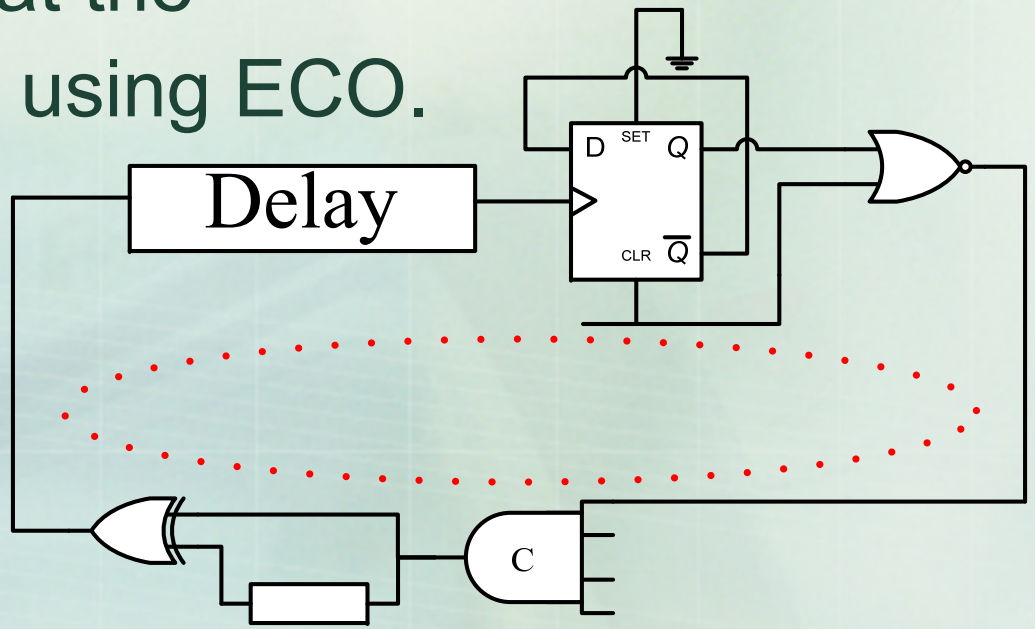
- Deliver the **required performance** under typical conditions.
- **Lower power** consumption.
- Help in **handling process variations**.
- Reduce **CAD** and Design **complexity**.

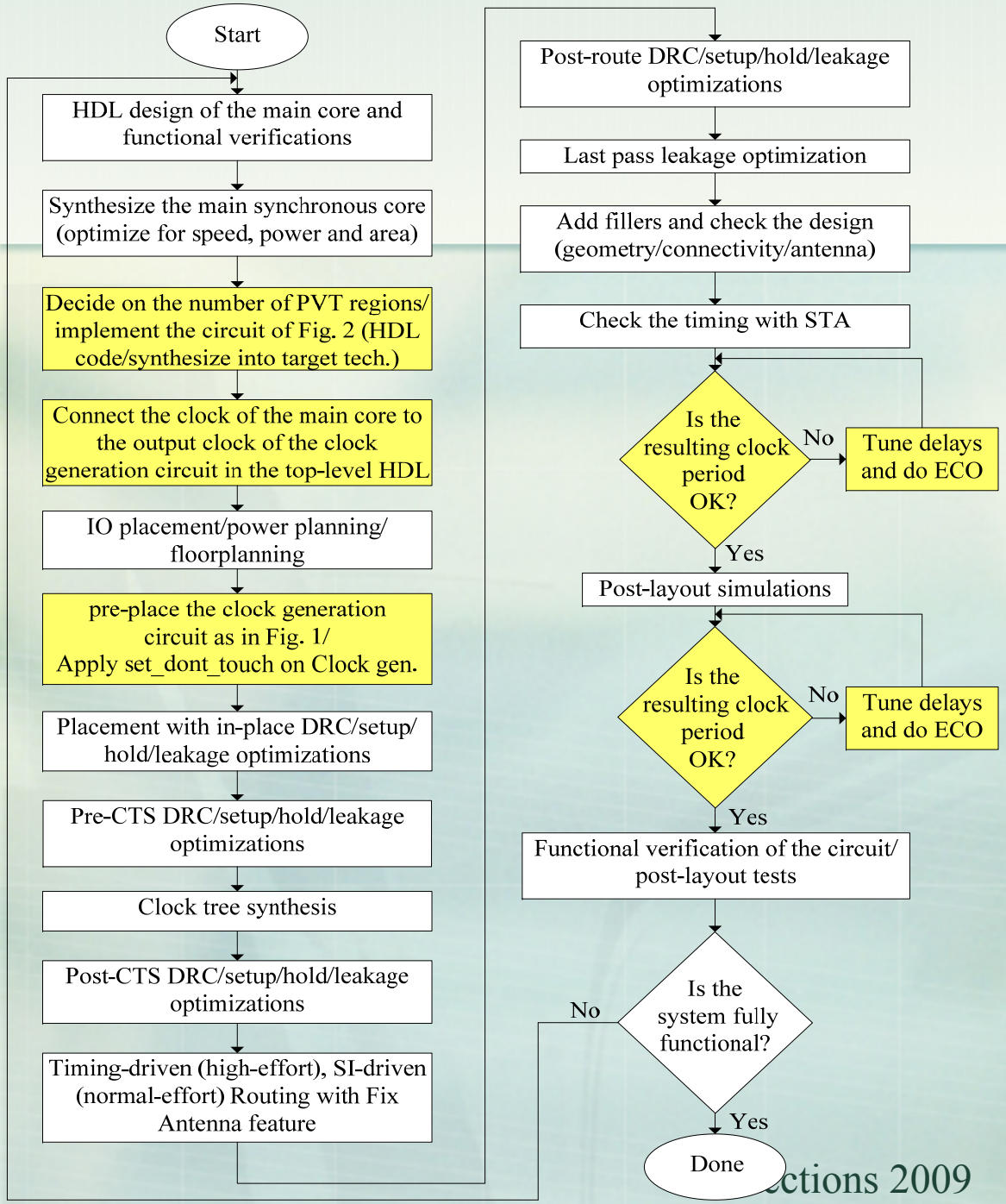
Clock Generation Schematic



Clock Generation Schematic

- The delay of the **loop** is tuned for the desired clock period.
- Fine-tune delays at the post-layout netlist using ECO.




Start
HDL design of the main core and functional verifications
Synthesize the main synchronous core (optimize for speed, power and area)
Decide on the number of PVT regions/ implement the circuit of Fig. 2 (HDL code/synthesize into target tech.)
Connect the clock of the main core to the output clock of the clock generation circuit in the top-level HDL
IO placement/power planning/ floorplanning
pre-place the clock generation circuit as in Fig. 1/ Apply set_dont_touch on Clock gen.
Placement with in-place DRC/setup/ hold/leakage optimizations
Pre-CTS DRC/setup/hold/leakage optimizations
Clock tree synthesis
Post-CTS DRC/setup/hold/leakage optimizations
Timing-driven (high-effort), SI-driven (normal-effort) Routing with Fix Antenna feature
Post-route DRC/setup/hold/leakage optimizations
Last pass leakage optimization
Add fillers and check the design (geometry/connectivity/antenna)
Check the timing with STA
Is the resulting clock period OK?
No
Tune delays and do ECO
Yes
Post-layout simulations
Is the resulting clock period OK?
No
Tune delays and do ECO
Yes
Functional verification of the circuit/ post-layout tests
Is the system fully functional?
No
Yes
Done

How to connect to the environment?

- Multiflop synchronizers
- Multiplexer recirculation techniques
- Use of first-in-first-out buffers between different clock domains
- Handshake techniques