Pipelined 32-bit Prefix Adder Workflow

Input Breakdown:

- Inputs are two 32-bit binary numbers: `A[31:0]` and `B[31:0]`.
- A carry-in (`C_in`, typically 0) may also be included.

Step 1: Generate and Propagate Signals:

- Compute **Generate (G)** and **Propagate (P)** signals for each bit: G[i]=A[i]⋅B[i](Carry generated by bit i)G[i] = A[i] \cdot B[i] \quad (\text{Carry generated by bit } i)G[i]=A[i]⋅B[i](Carry generated by bit i) P[i]=A[i]⊕B[i](Carry propagated by bit i)P[i] = A[i] \oplus B[i] \quad (\text{Carry propagated by bit } i)P[i]=A[i]⊕B[i](Carry propagated by bit i)

Step 2: Pipeline Stage 1 - Initial Combination:

- Group adjacent bits to compute intermediate generate and propagate signals: G′[i]=G[i]+(P[i]⋅G[i−1])G'[i] = G[i] + (P[i] \cdot G[i-1])G′[i]=G[i]+(P[i]⋅G[i−1]) P′[i]=P[i]⋅P[i−1]P'[i] = P[i] \cdot P[i-1]P′[i]=P[i]⋅P[i−1]
- Store results in pipeline registers for further processing.

Step 3: Pipeline Stage 2 - Group Propagation:

- Combine results from Stage 1 to compute wider groups (e.g., 4-bit or 8-bit blocks): G″[i]=G′[i]+(P′[i]⋅G′[i−2])G''[i] = G'[i] + (P'[i] \cdot G'[i-2])G″[i]=G′[i]+(P′[i]⋅G′[i−2]) P″[i]=P′[i]⋅P′[i−2]P''[i] = P'[i] \cdot P'[i-2]P″[i]=P′[i]⋅P′[i−2]
- Use pipeline registers to hold these intermediate values.

Step 4: Final Carry Calculation:

- Continue combining blocks in subsequent pipeline stages until all carries (`C[i]`) are computed.

Step 5: Compute the Sum:

- Calculate the sum for each bit using the propagate signals and computed carries: S[i]=P[i]⊕C[i−1]S[i] = P[i] \oplus C[i-1]S[i]=P[i]⊕C[i−1]

Output Assembly:

- Combine the sum bits (`S[31:0]`) for the final 32-bit sum.
- The carry-out from the final stage (`C[32]`) indicates overflow.

Pipeline Stages:

- Intermediate results (`G`, `P`, and `C`) are stored in registers between stages to ensure high throughput.