Resource Optimization Techniques for Macrocell-Based CPLDs

Agenda

- Objectives
- MAX Device Architecture
- Resource Optimization Techniques
- Conclusion
- Appendix

Agenda

- Objectives
- MAX Device Architecture
- Resource Optimization Techniques
- Conclusion
- Appendix

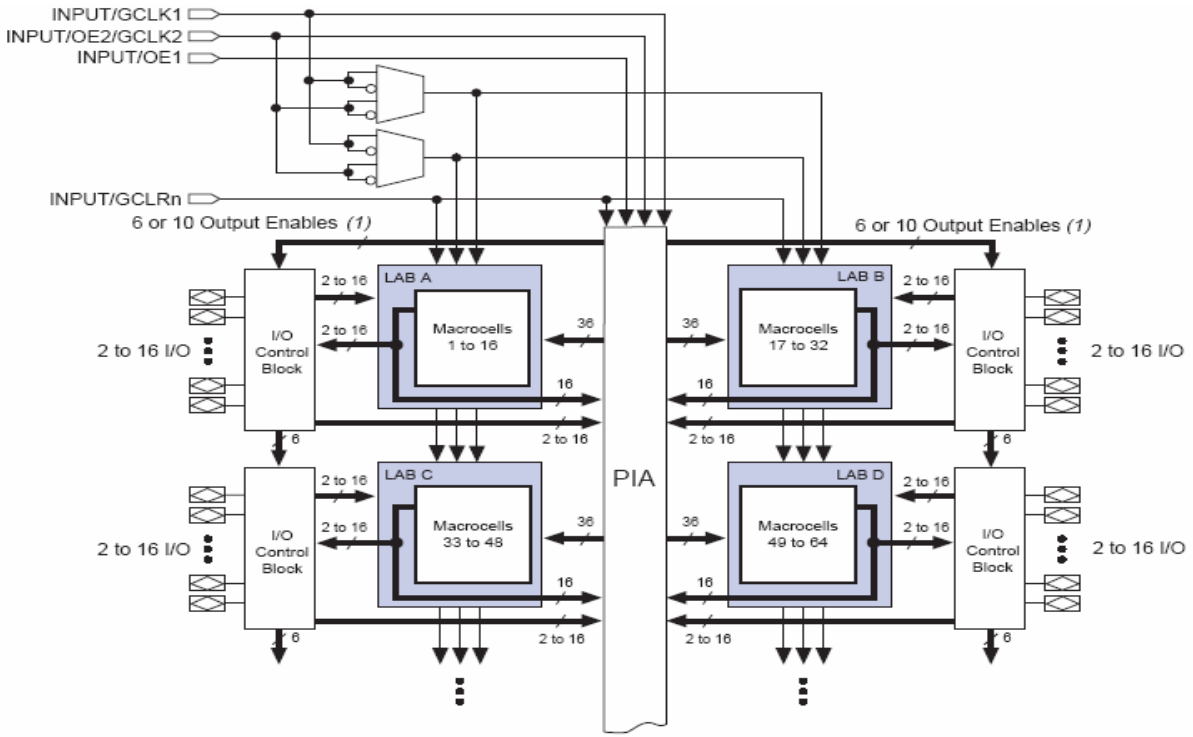
Objectives

- To take full advantage of the architectures of the MAX3000 and MAX7000 devices by
  - Optimizing macrocell utilization and/or
  - Optimizing routing resource utilization.
- To resolve the macrocell usage issues and routing issues.

Agenda

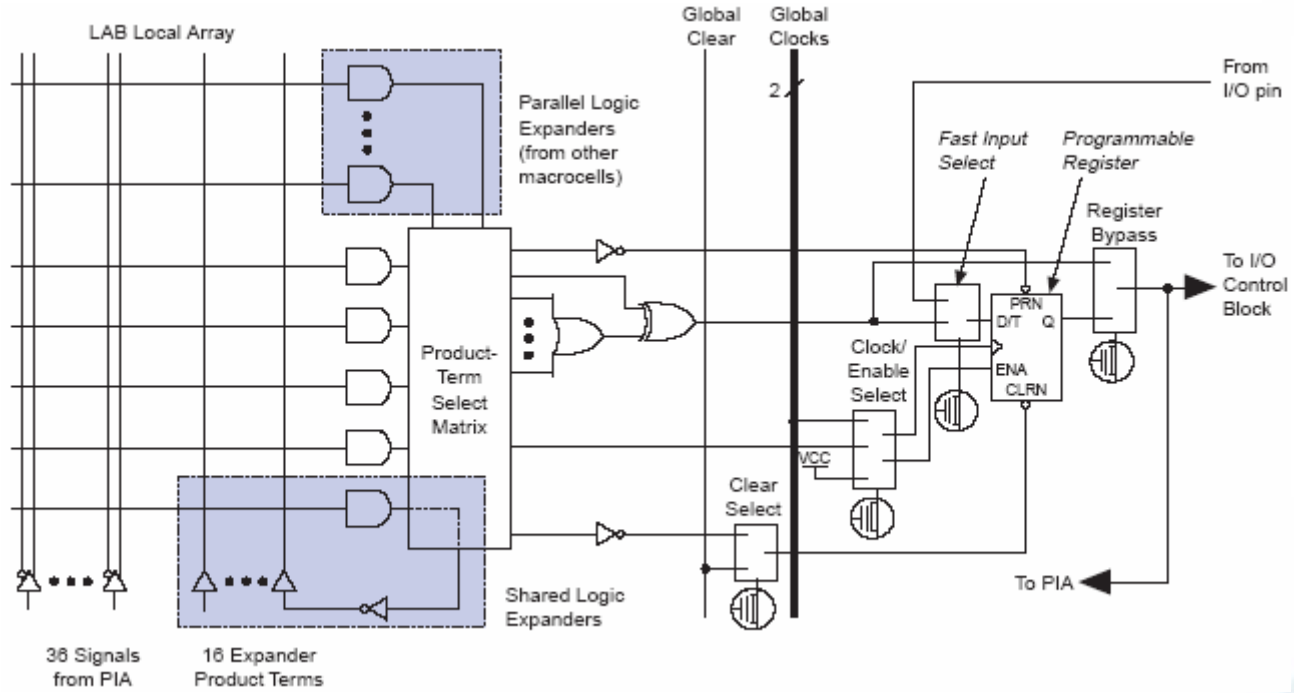
- Objectives
- MAX Device Architecture
- Resource Optimization Techniques
- Conclusion
- Appendix

MAX Device Architecture



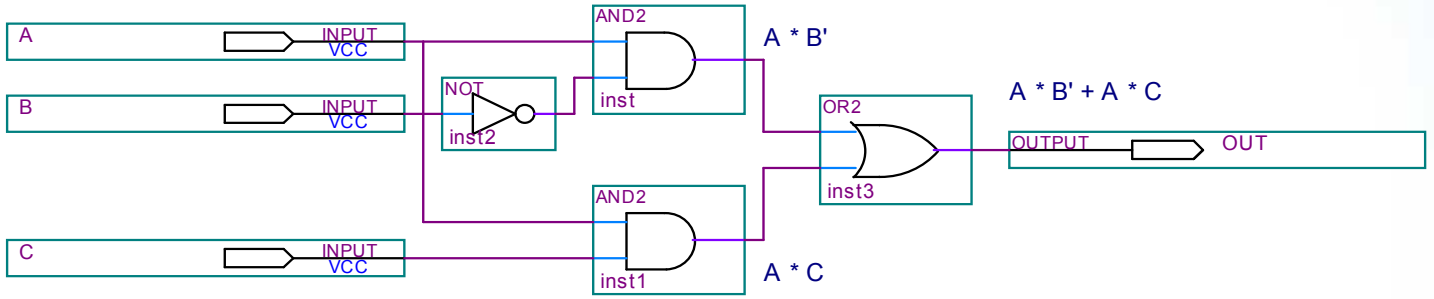
- Logic Array Blocks(LABs) & Macrocells
- Programmable Interconnect Array(PIA)
- I/O Control Blocks
- Dedicated Global Inputs

Macrocell Architecture

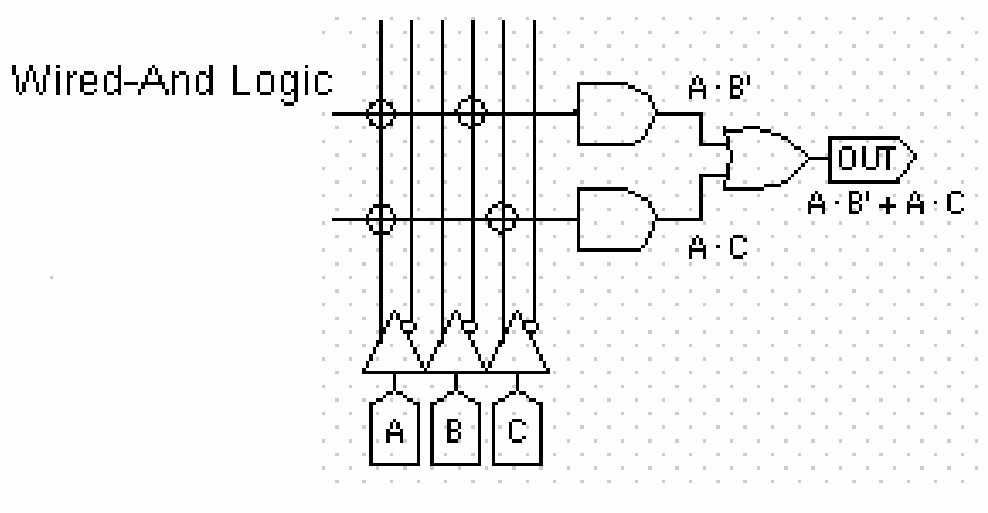


- LAB Local Logic Array (Five Programmable Product Terms)
- Product-Term Select Matrix
- Programmable Register

Product-Term Line Usage Example



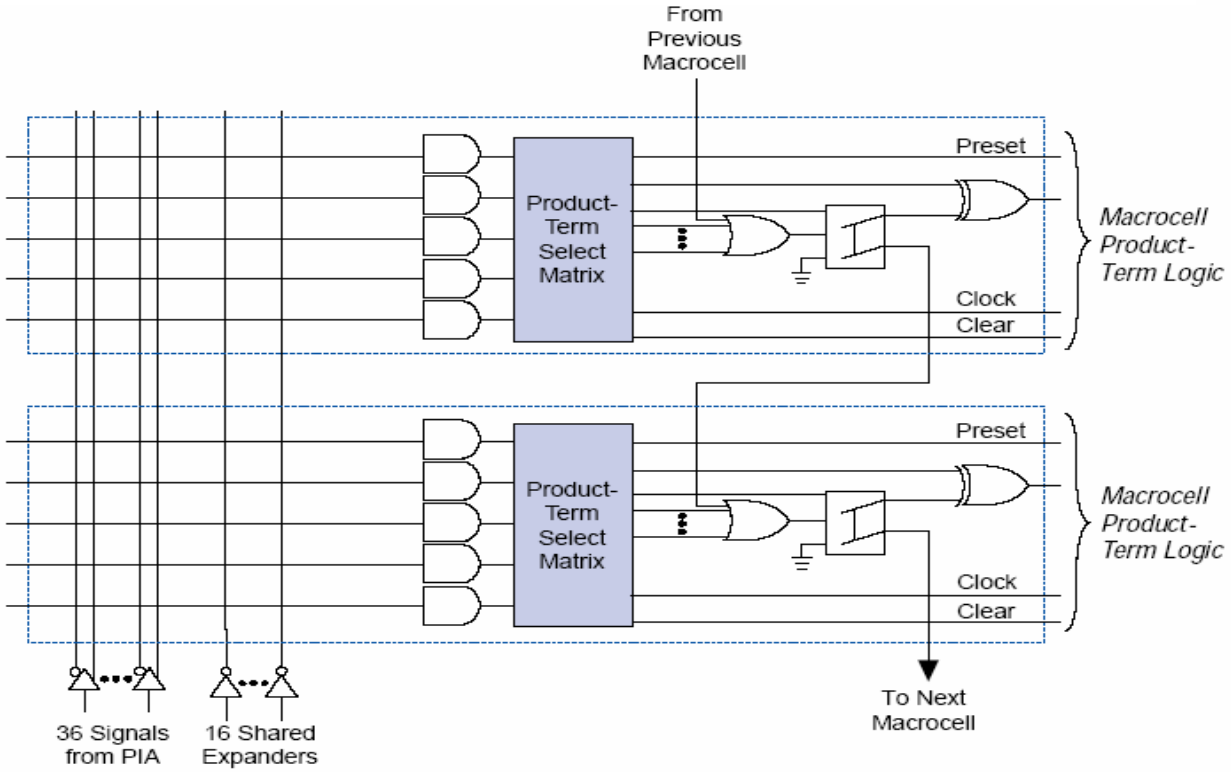
OUT = A * B’ + A * C ⬇ Implemented in MAX Macrocell



Expander Product Terms

- Additional Product Terms provided by the other macrocells in the same LAB.
- Used to implement more complex logic functions.
- Include
  - Parallel Expanders
  - Sharable Expanders

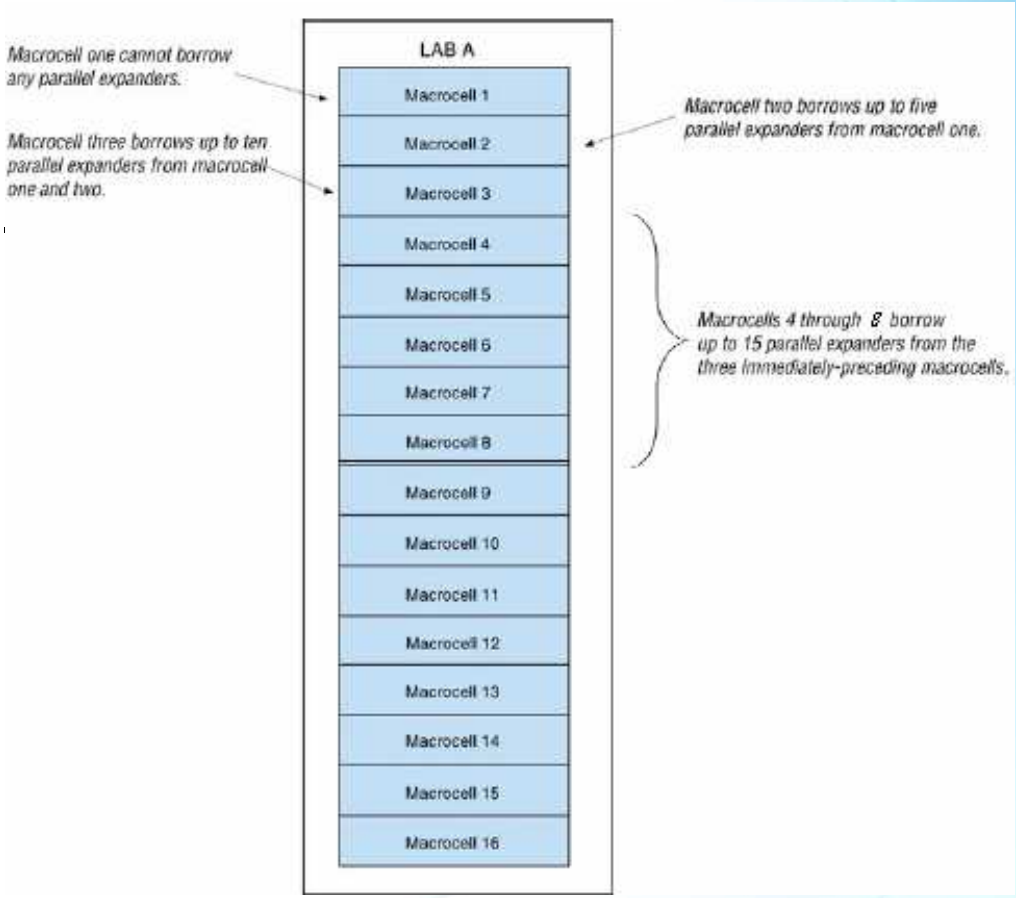
Parallel Expanders



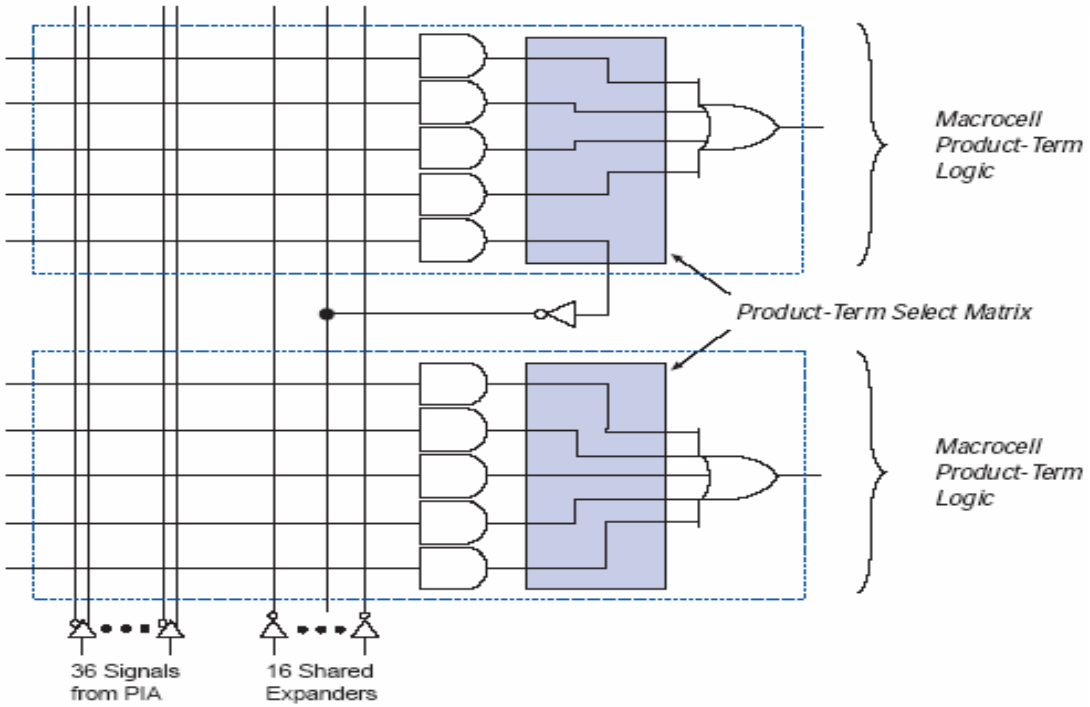
- Product terms borrowed from the neighboring macrocells.
- Up to 15 parallel expanders can be borrowed from three neighboring macrocells.
- The macrocell with parallel expanders lent cannot implement other logic.

Parallel Expanders Borrowing Rule

- Two groups of 8 macrocells within each LAB (macrocells 1 through 8 and 9 through 16) form two chains to lend or borrow parallel expanders.
- A macrocell borrows parallel expanders from lower-numbered macrocells.
- Within each group of 8, the lowest-numbered macrocell can only lend parallel expanders and the highest-numbered macrocell can only borrow them.



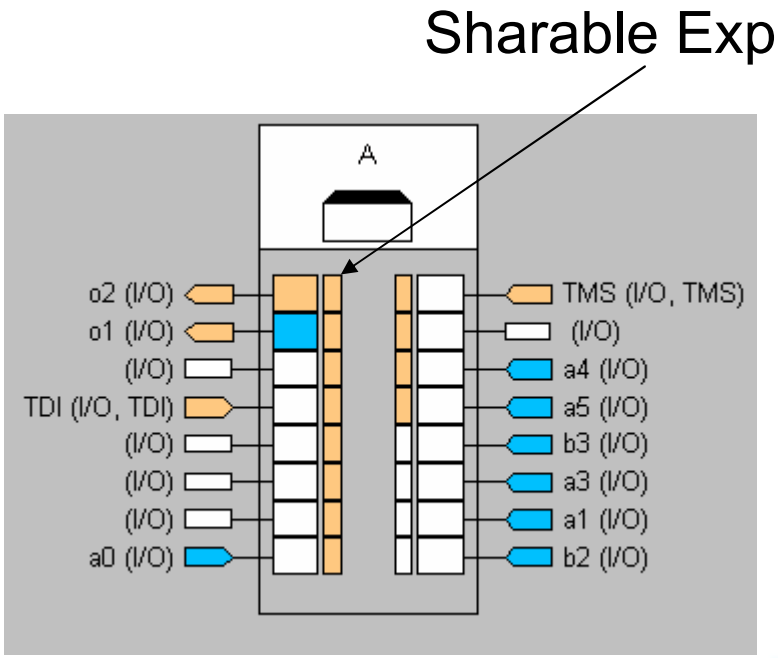
Sharable Expanders



- 16 Product terms feeding back into the logic array from the 16 macrocells in an LAB, one from each macrocell.
- Sharable expanders be shared by any or all macrocells in an LAB.
- The macrocell providing a sharable expander can still implement other logic.

Sharable Expanders in Timing Closure Floorplan

- The sharable expander and the macrocell where it is sourced can be used simultaneously.



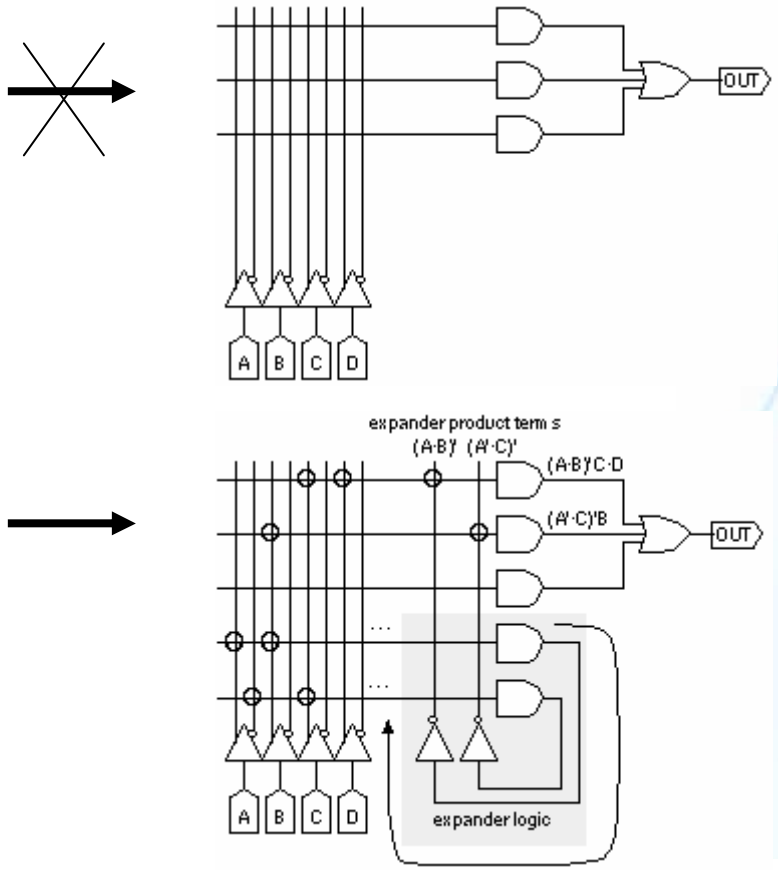
Sharable Expanders Usage Example

- Assume there are only three product terms in each macrocell. Consider the function

*OUT = A'*C*D+B'*C*D+A*B+B*C'*

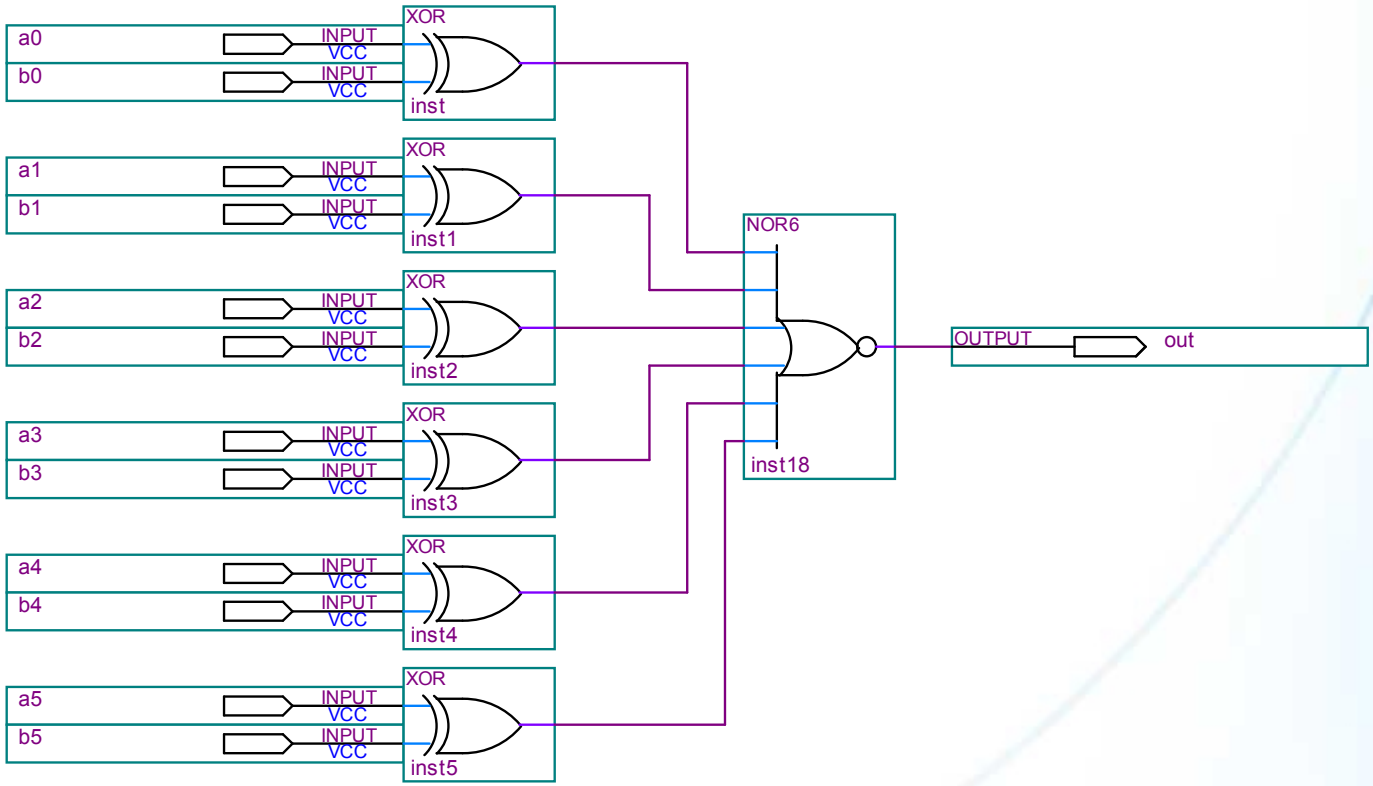
*= (A'+B')*C*D+(A+C')*B*

*= (A*B)'*(C*D) + (A'*C)'*B*



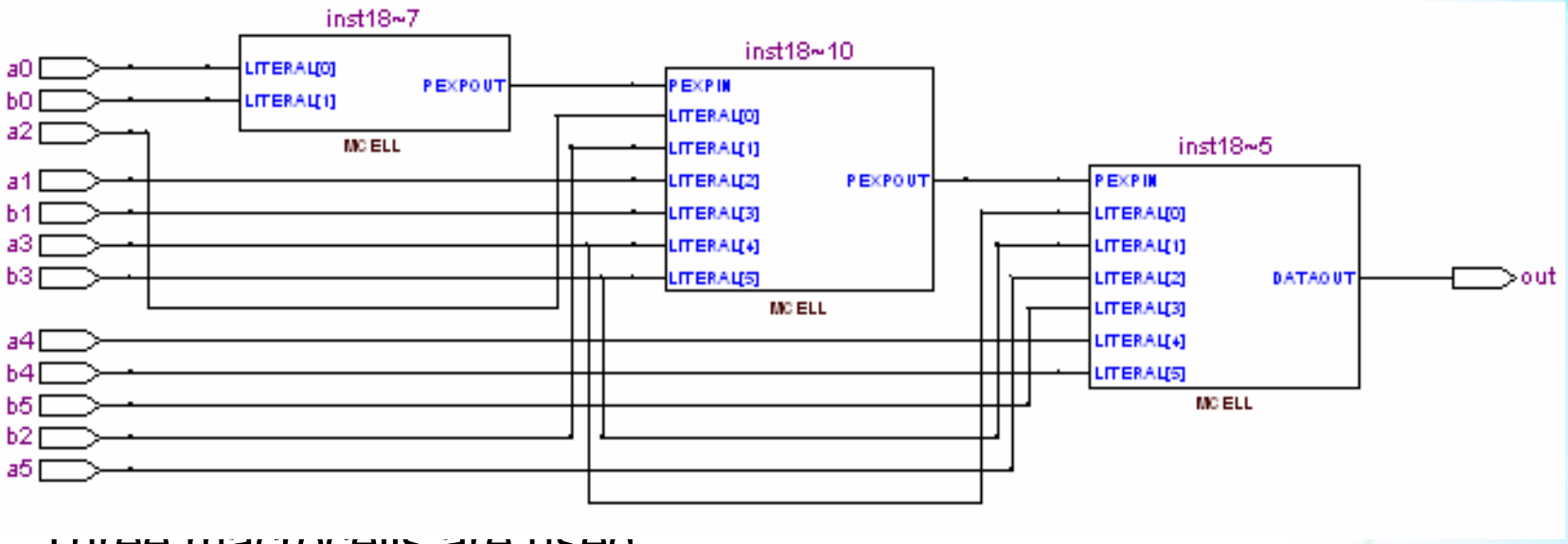
Example Resource Usage Compare:
Parallel Expanders vs. Sharable Expanders

- Consider the following Quartus II design implemented in MAX7000AE device.



Parallel Expanders Implementation

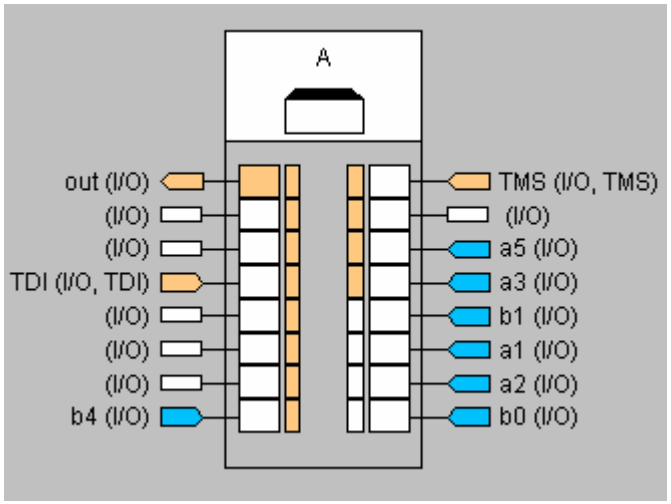
- Set “Auto Parallel Expanders” as On.

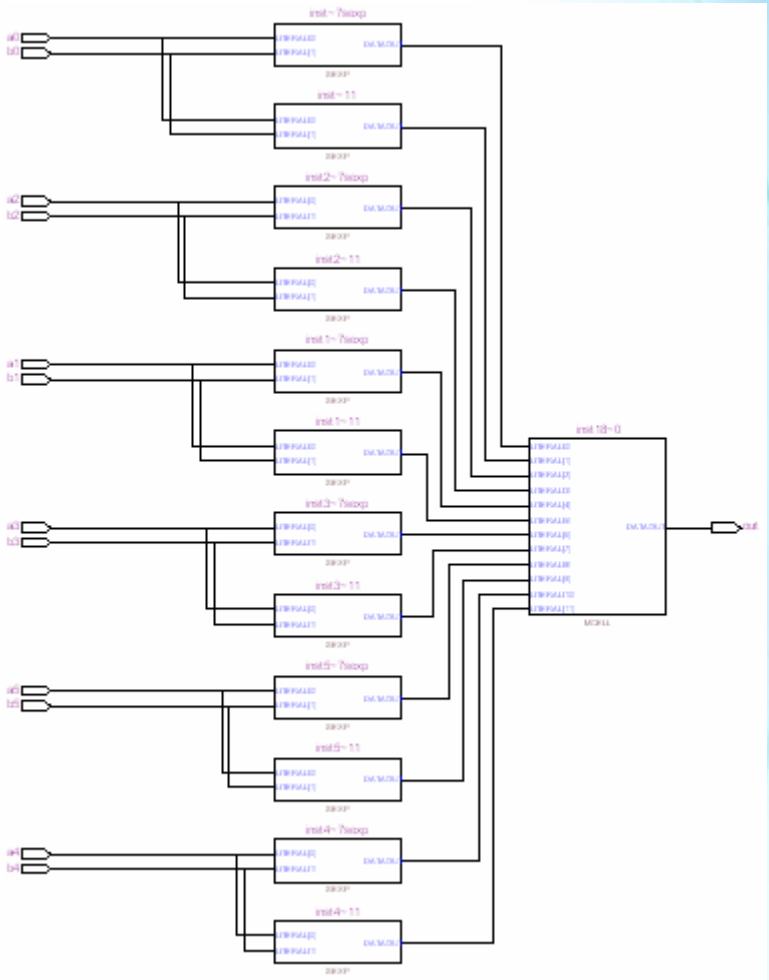


- Three macrocells are used.
- Two are used to provide parallel expanders(PEXPOUT).

Sharable Expanders Implementation

- Only one macrocell is used.
- 12 sharable expanders.





User Control of Selection: Parallel Expanders or Sharable Expanders

- Users can direct Quartus II to use or avoid using parallel expanders by setting the “Auto parallel expanders” option On or off.
- Using or not using sharable expanders cannot be controlled by users, only decided by Quartus II.
- Quartus II can balance the resource and timing performance of the user design by automatically selecting the uses of sharable expanders.

Agenda

- Objectives
- MAX Device Architecture
- Resource Optimization Techniques
- Conclusion
- Appendix

Resource Optimization Techniques

- To optimize resource usage and avoid no-fit issues by
  - Minimizing macrocell usage and/or
  - Minimizing routing resource usage.
- Basic settings to achieve the above two aspects
  - Set “Synthesis Optimization Technique” as “Area”
  - Use dedicated inputs for global control signals
    - Global register control resources can only be accessed through the dedicated input pins.
    - Can save logic array resources and interconnect resources.
  - Remove redundant logic
    - Ignore CARRY/CASCADE/LCELL/GLOBAL/SOFT buffers.
    - Power-Up Don’t Care / Remove Duplication Logic / Auto Resource Sharing / State Machine Processing, etc.

Basic Setting Optimization Case: EPM3512 Device

- Simple issue description
  - The following errors appear during fitting
    - Error: Can't place node "<name>" of type <type>
  - Macrocell usage is 349/512
- Solution
  - A routing issue.
  - Changing the pin mapping of a clock signal to GCLK resolves this issue.

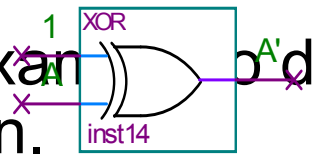
Techniques for Optimizing Macrocell Usage

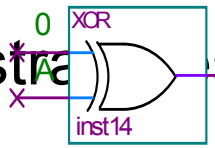
- Basic setting options
- Turn on “Allow XOR Gate Usage”
- Turn off “Auto Parallel Expanders”
- Optimize source codes

Turn on “Allow XOR Gate Usage”

- The usage of the XOR gate in the macrocell is to implement programmable inversion to the logic function.
- The function of XOR gate

- An example demonstrates resource saving by using this option.





*F = A*B’+A*C’+A*D’+A’*C*D*

can not be implemented with three product term lines while

F’ = A*B*C*D+A’*D’+A’*C’

can.

Turn off “Auto Parallel Expanders”

- When “Auto Parallel Expanders” is on,
  - Quartus II tends to use more parallel expanders instead of more sharable expanders.
  - the amount of the macrocells required usually increases.

Optimize source codes

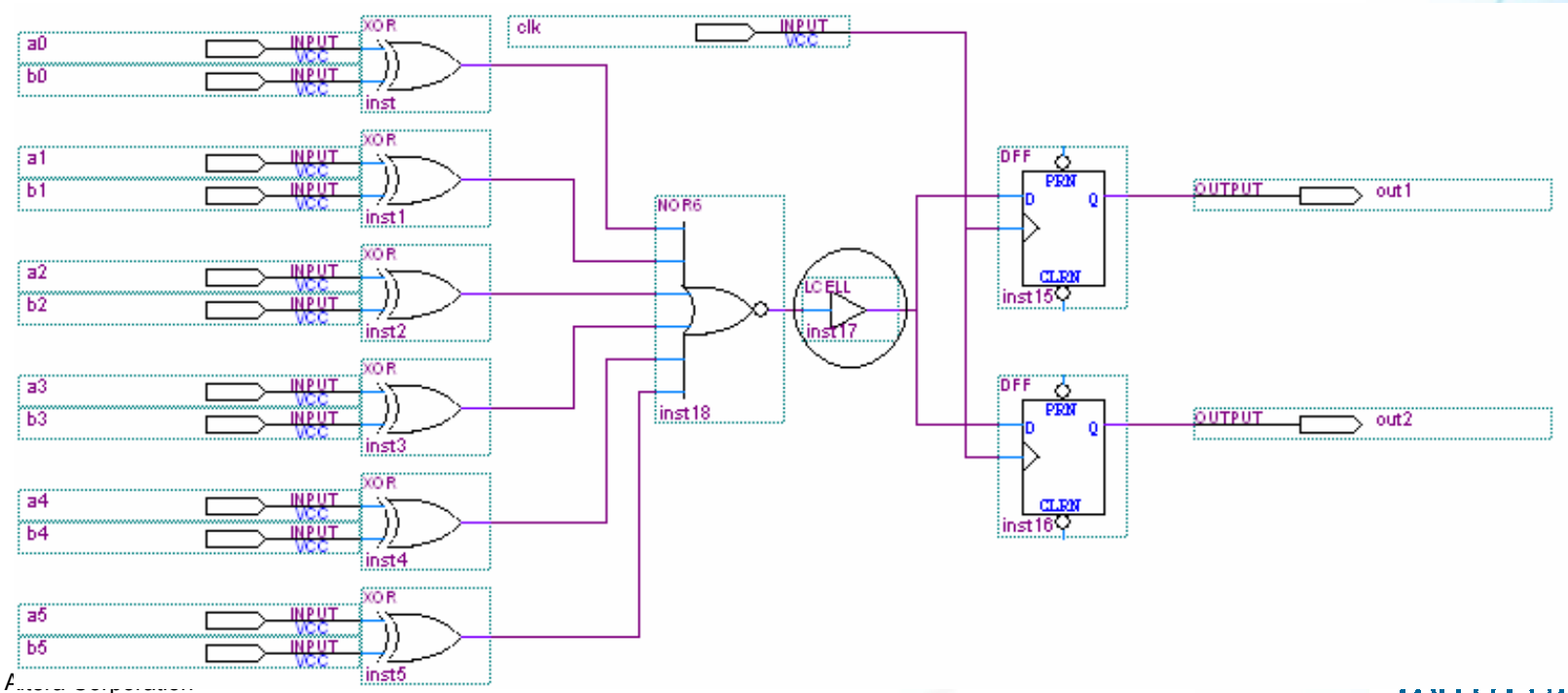
- The most frequently used method to reduce the macrocell usage.
- Some Tips
  - Use D Flipflops instead of Latches;
  - Use asynchronous control signals instead of synchronous ones;
- It is case by case.

Techniques for Optimizing Routing Usage

- Routing issues usually occur on some routing areas in PIA or in LAB.
- Techniques
  - Basic setting options
  - Turn on “Auto Logic Cell Insertion”
  - Turn off “Auto Parallel Expanders”
  - Reduce fan-in per macrocell
  - Optimize source codes

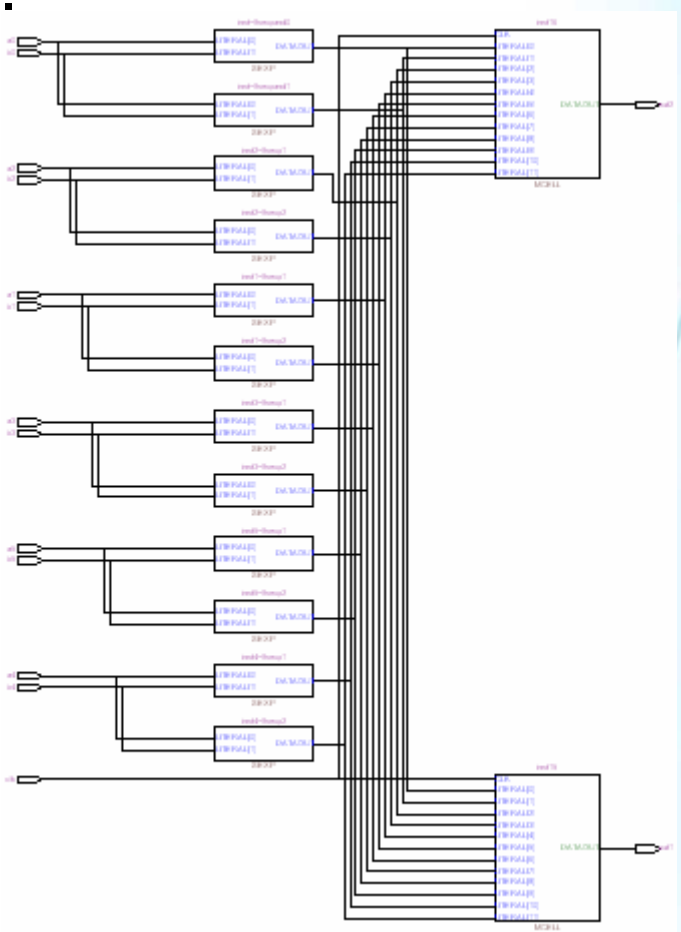
Turn on “Auto Logic Cell Insertion”

- Inserting logic cells in some complex module reduces fan-in and shared expanders used per macrocell, increasing routability.
- To demonstrate this option, consider the function



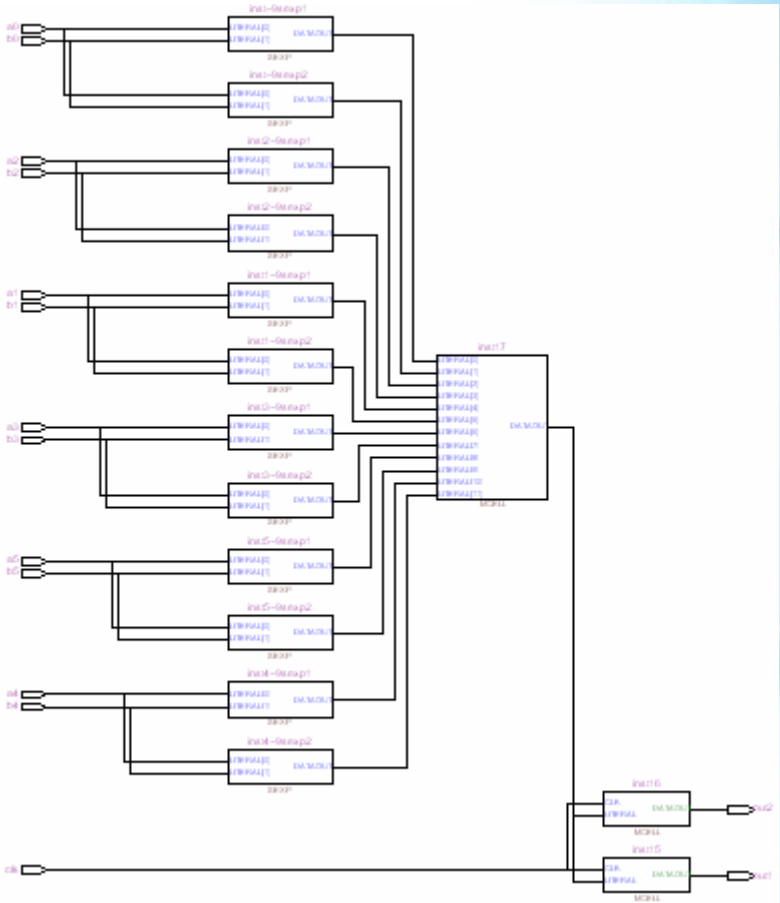
Logic Cell Insertion Example

- Turn on “Ignore LCELL Buffers”.
  - Two macrocells used;
  - 12 fan-ins per macrocell.



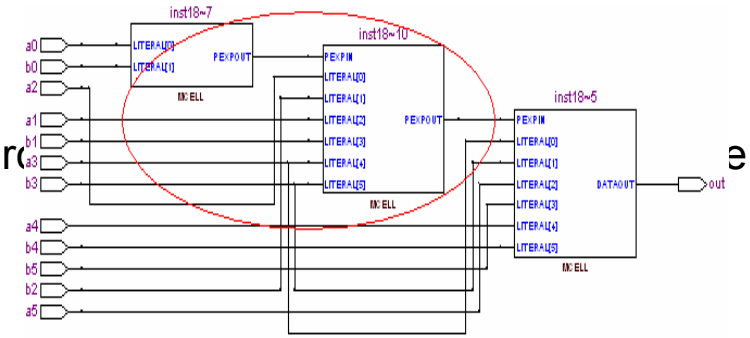
Logic Cell Insertion Example(Cont.)

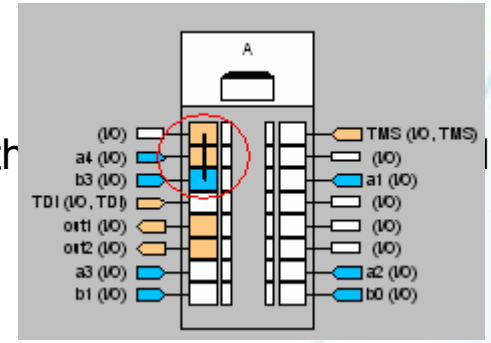
- Turn off “Ignore LCELL Buffers”.
  - Three macrocells used;
  - 4.67 fan-ins per macrocell.



Turn off “Auto Parallel Expanders”

- When “Auto Parallel Expanders” is on,
  - the fitting flexibility for each macrocell will be decreased.
    - The location of macrocells in an LAB is limited to borrow parallel expanders.
    - Chains of parallel expanders tend to be created to implement complex logic.
  - r[illegible]e LAB with[illegible].





Reduce fan-in per macrocell

- Generally set the “Maximum Fan-in Per Macrocell” option to 40% - 60%.
  - Quartus II distributes the inputs of a single macrocell to multiple macrocells.
  - Distributing the inputs of macrocells across LABs reduces routing congestion for LABs with high fan-in.

Routing Optimization Case: EPM3256 Device

- Simple issue description
  - The following errors appear during fitting
    - Error: Can't route source node “<name>" of type max_mcell to the OE port of destination node “<name>" of type max_io
    - Error: Can't route source node “<name>” of type max_mcell to destination node “<name>" of type max_mcell
    - Error: Can't place node “<name>" of type max_mcell
  - Macrocell usage is 238/256
- Solution
  - Typical error messages indicating that not enough routing resources in PIA and LABs for the design.
  - The "Area" Synthesis Optimization option, Auto Parallel Expanders to off and Maximum Fan-in Per Macrocell to 50% can resolve this routing issue.

Optimize source codes

- Some Typical methods
  - Use asynchronous control signals instead of synchronous ones;
  - Manually insert logic cells.
    - break down the complex logic with high fan-in.
    - separate complex logic from its high fan-out.
- Also case by case.
- Generally, if routing issue occurs while macrocell utilization is low, the Quartus II routing optimization options are enough to fix it. No need to modify source codes.

Agenda

- Objectives
- MAX Device Architecture
- Resource Optimization Techniques
- Conclusion
- Appendix

Conclusion(1): Routing Resource

- The routing resource issues are easy to fix if the macrocell usage in the design is low.
- Use the following settings
  - Basic setting options
  - Turn on “Auto Logic Cell Insertion” (default)
  - Turn off “Auto Parallel Expanders”
  - set the “Maximum Fan-in Per Macrocell” option to 40% - 60%.

Conclusion(2): Macrocell

- Macrocell usage issues are more difficult to fix than routing ones.
- Use the following settings
  - Basic setting options
  - Turn on “Allow XOR Gate Usage”
  - Turn off “Auto Parallel Expanders”
- In many cases designers have to optimize source codes.
- Some setting options may have adverse effects to the macrocell usage due to the improper coding style.

Agenda

- Objectives
- MAX Device Architecture
- Resource Optimization Techniques
- Conclusion
- Appendix

Appendix: Referred Document

- For the architecture of MAX3000A Devices, please refer to

  MAX 3000A Programmable Logic Device Family Data Sheet, at
  http://www.altera.com/literature/ds/m3000a.pdf.

- For the recommended design guidelines, please refer to

  Section II. Design Guidelines of Quartus II Handbook Volume 1: Design & Synthesis, at
  http://www.altera.com/literature/hb/qts/qts_qii5v1_02.pdf.

- For the resource optimization techniques for macrocell-based CPLD devices, please refer to
  - *Resource Utilization Optimization Techniques (Macrocell- Based CPLDs) at Chapter 8. Area & Timing Optimization of Quartus II Version Handbook Volume 2: Design Implementation & Optimization, at*
    http://www.altera.com/literature/hb/qts/qts_qii52005.pdf.
  - *In Quartus II, click Tools menu -> Advisors -> Resource Optimization Advisor.*