Search Engine

For SVFIG

Dr. C. H. Ting
Offete Enterprises

December 16, 2006

Summary

- Early Search Engine Design
- Challenges of Search Engines
- Complete Search Engine Design
- Components of Search Engine
- Implementation
- Conclusion

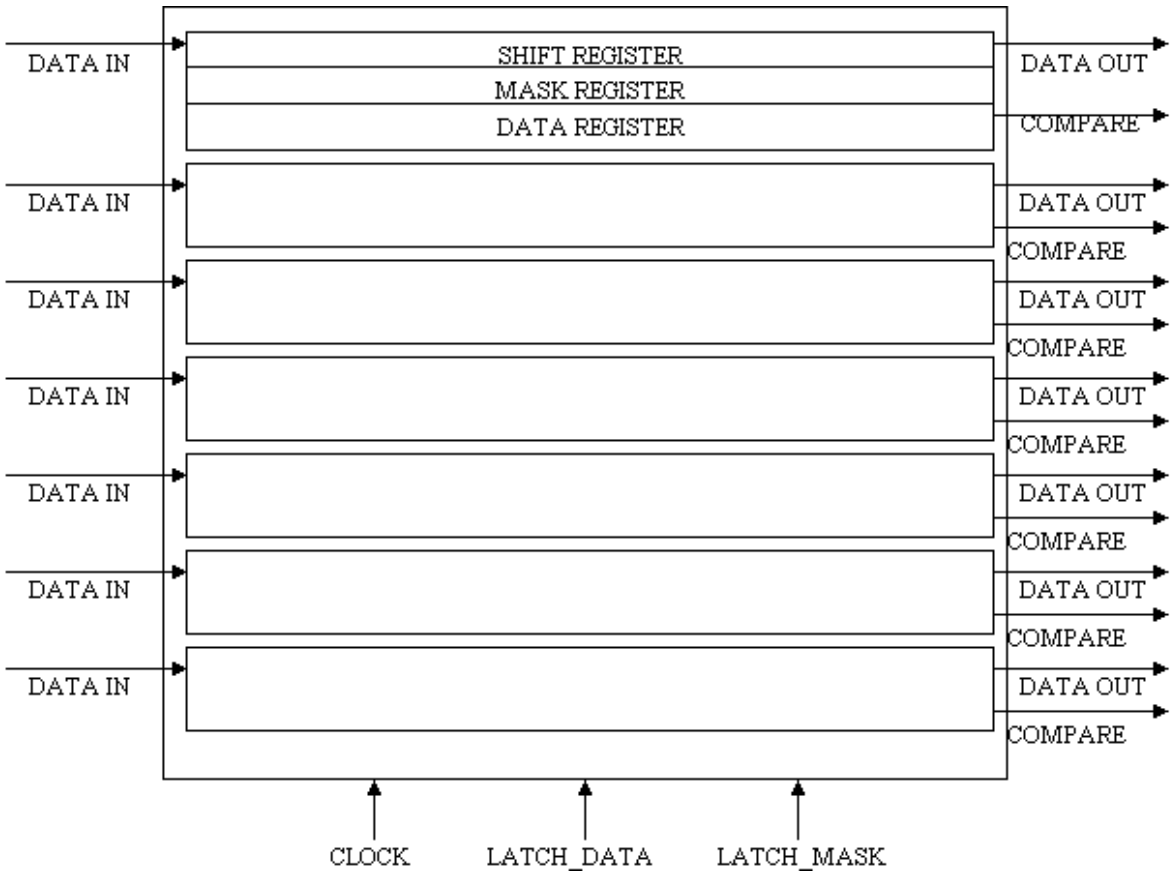
Early Search Engine Design

- **DC7 Search Engine**
  - **Seven 70-bit pipelined comparators for parallel data searching, prototyped by Orbit Semiconductor.**
- **Logic Equation**

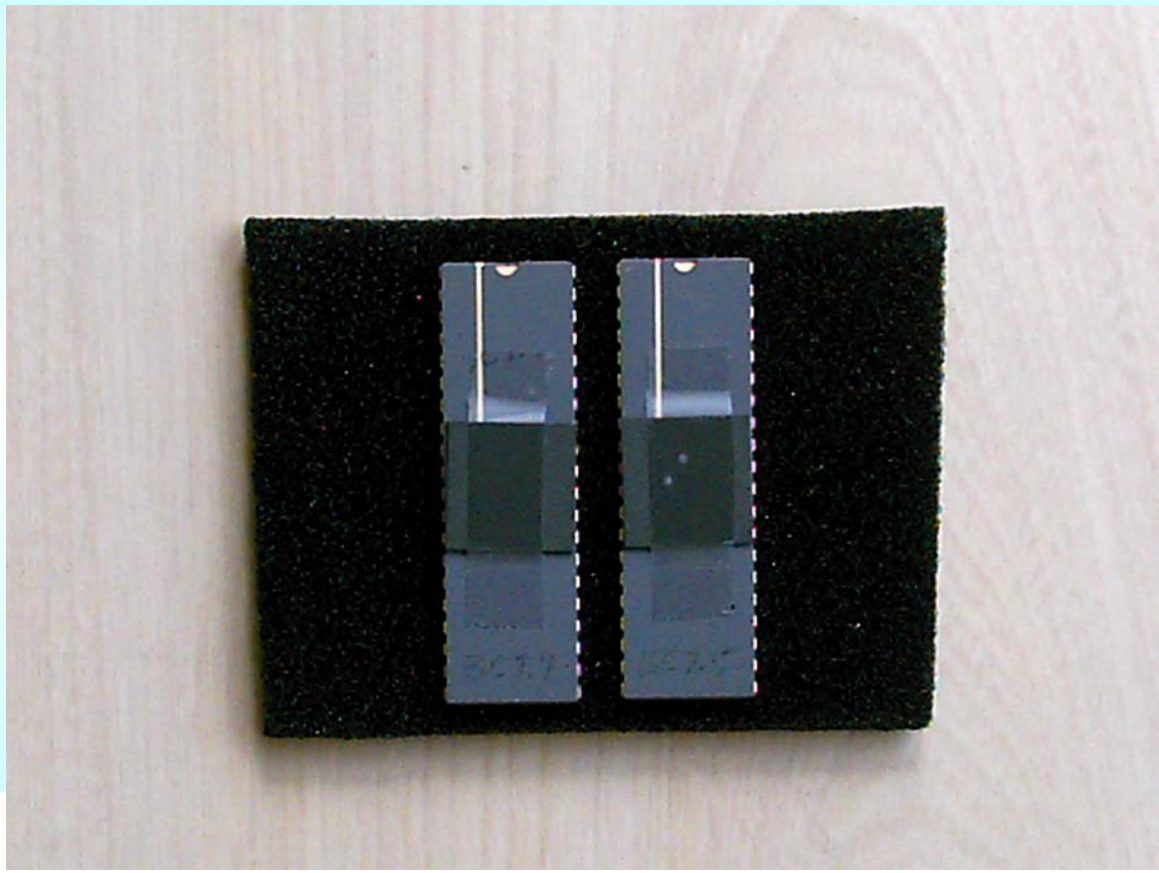
  Compare <= ‘1’ when

  ((shifter xor data) and mask) = 0 else ‘1’;

DC7 Search Engine



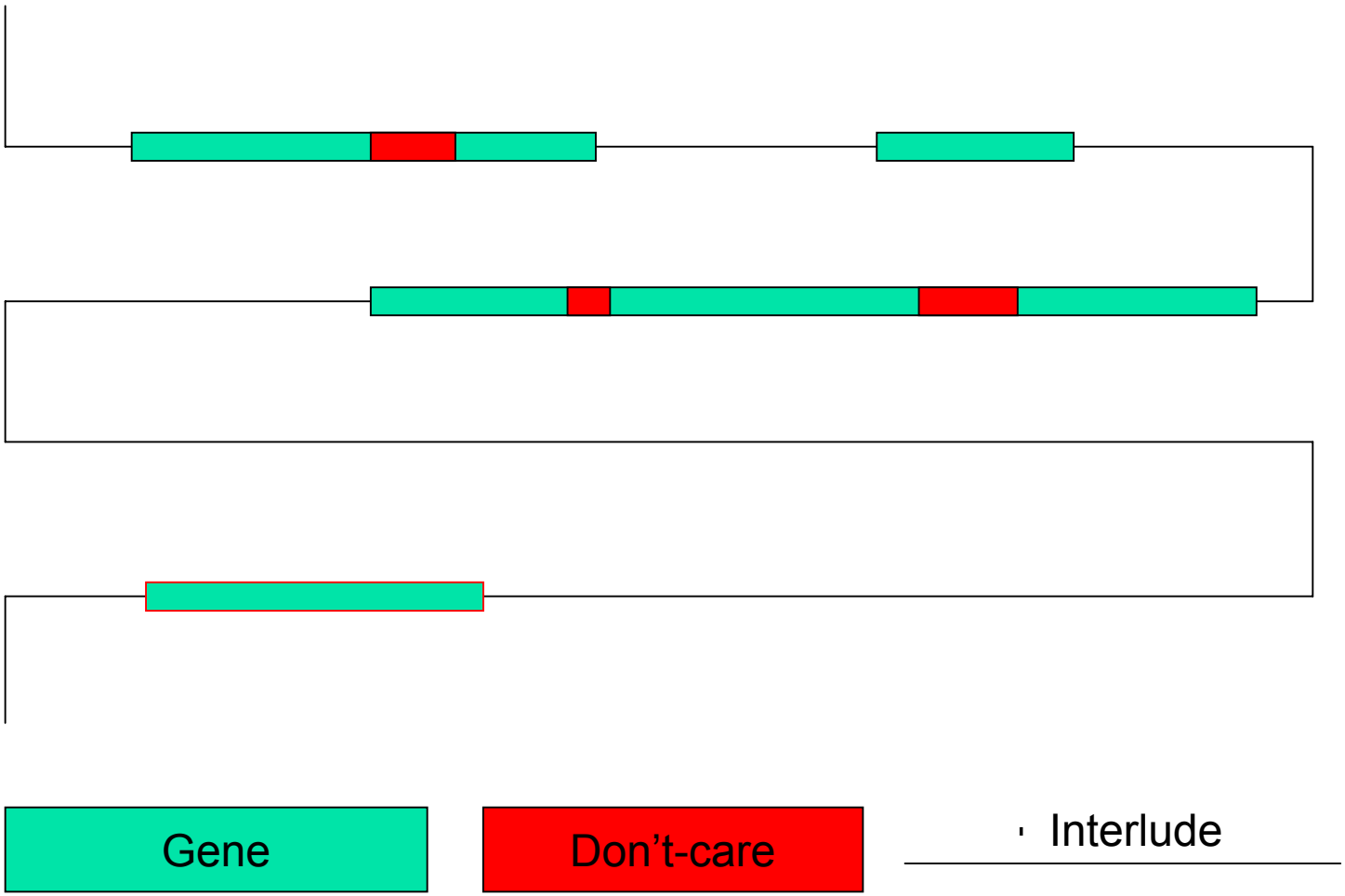
DC7 Search Engine



Challenges of a Modern Search Engine

- Many keywords—AND relation
- Keywords may contain don't-care characters—Maskable characters
- Keywords may have spatial relationships—Variable interludes
- Keywords may have alternatives—OR relation

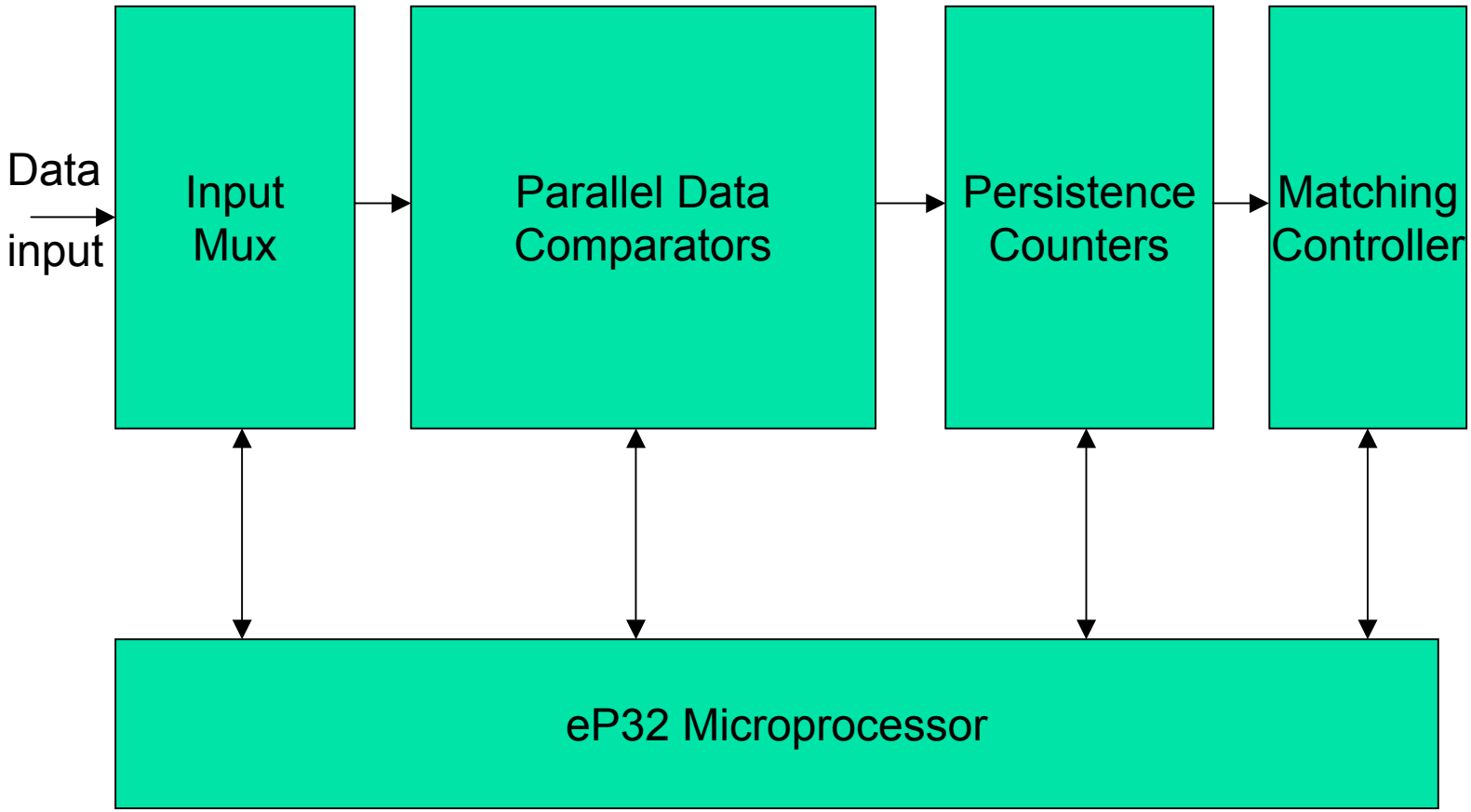
Challenges of a Gene Search Engine



Complete Search Engine Design

- Input Multiplexers
- Data Comparators
- Persistence Counters
- Matching Controller
- eP32 Microprocessor

Components of Search Engine



Input Multiplexers

- 32x32 Crossbar Input Multiplexers
- Any input bit stream can be directed to any Data Comparator
- Parallel input streams are used to load Data and Mask Registers
- A single serial input stream is used to distribute input data to all comparators

Data Comparators

- 32 200-bit Shifters with matching Data Registers and Mask Registers
- Controlling Signals
  - Master Clock
  - SHIFT
  - LOAD_DATA
  - LOAD_MASK
- COMPARE output

Persistence Counters

- **32 Persistence Counters to stretch the COMPARE signals from each comparator**
- **Persistence allows matching of several data segments with variable interludes among them**

Matching Controller

- Flexible OR-AND structure allowing persistent compare signals to be OR'ed and AND'ed together
- OR'ed compare signals are AND'ed to produce final HIT signal
- Locations of 16 Final HIT signals are logged for software analysis

Matching Controller

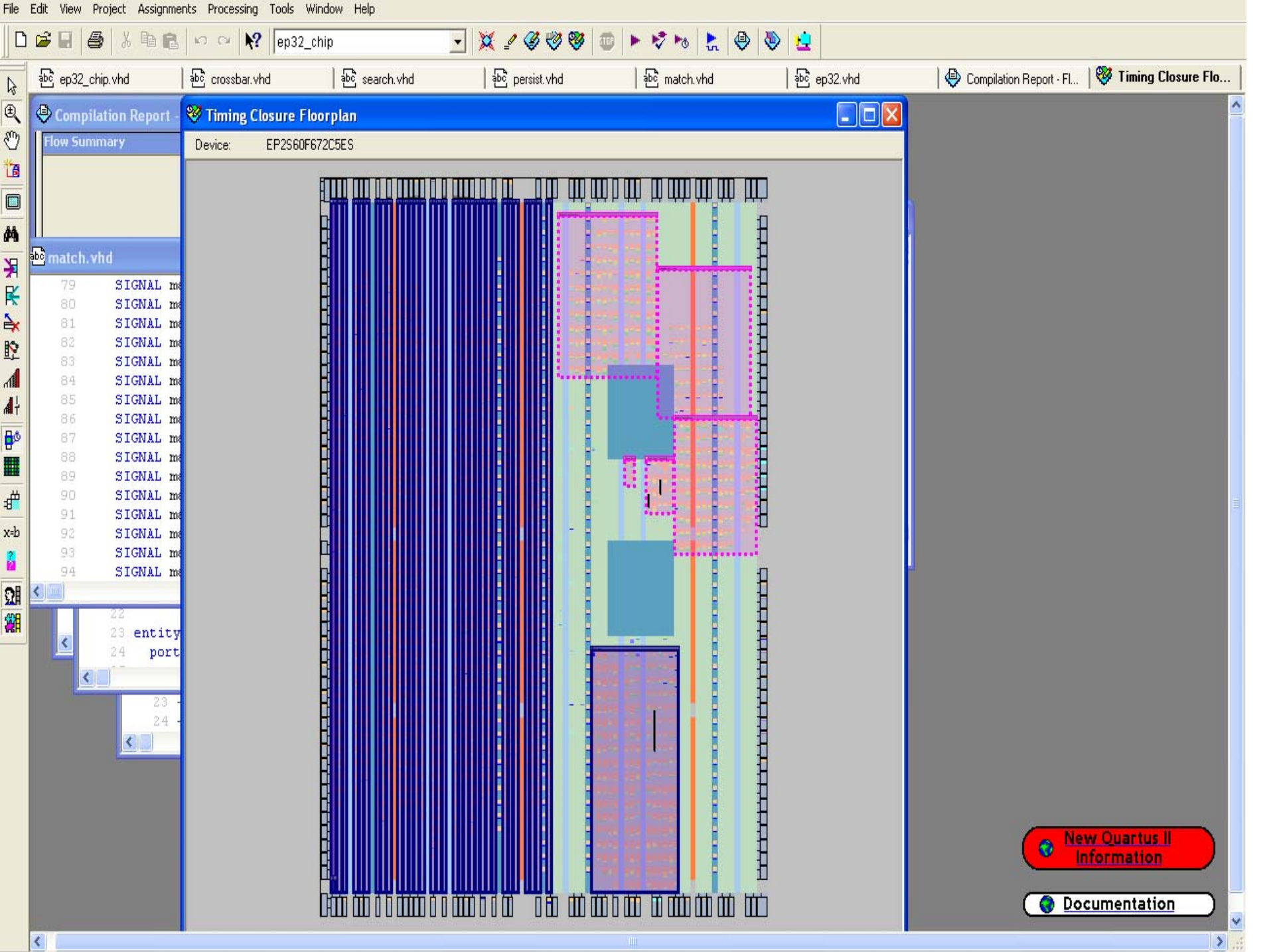
- **Match_Command Register issues SHIFT, LOAD_DATA and LOAD_MASK commands to Data Comparator**
- **Match_Parallel Register routes data to Input Mux in parallel**
- **Match_Serial Register routes data to Input Mux in a series of 32 clocks**

eP32 Microprocessor

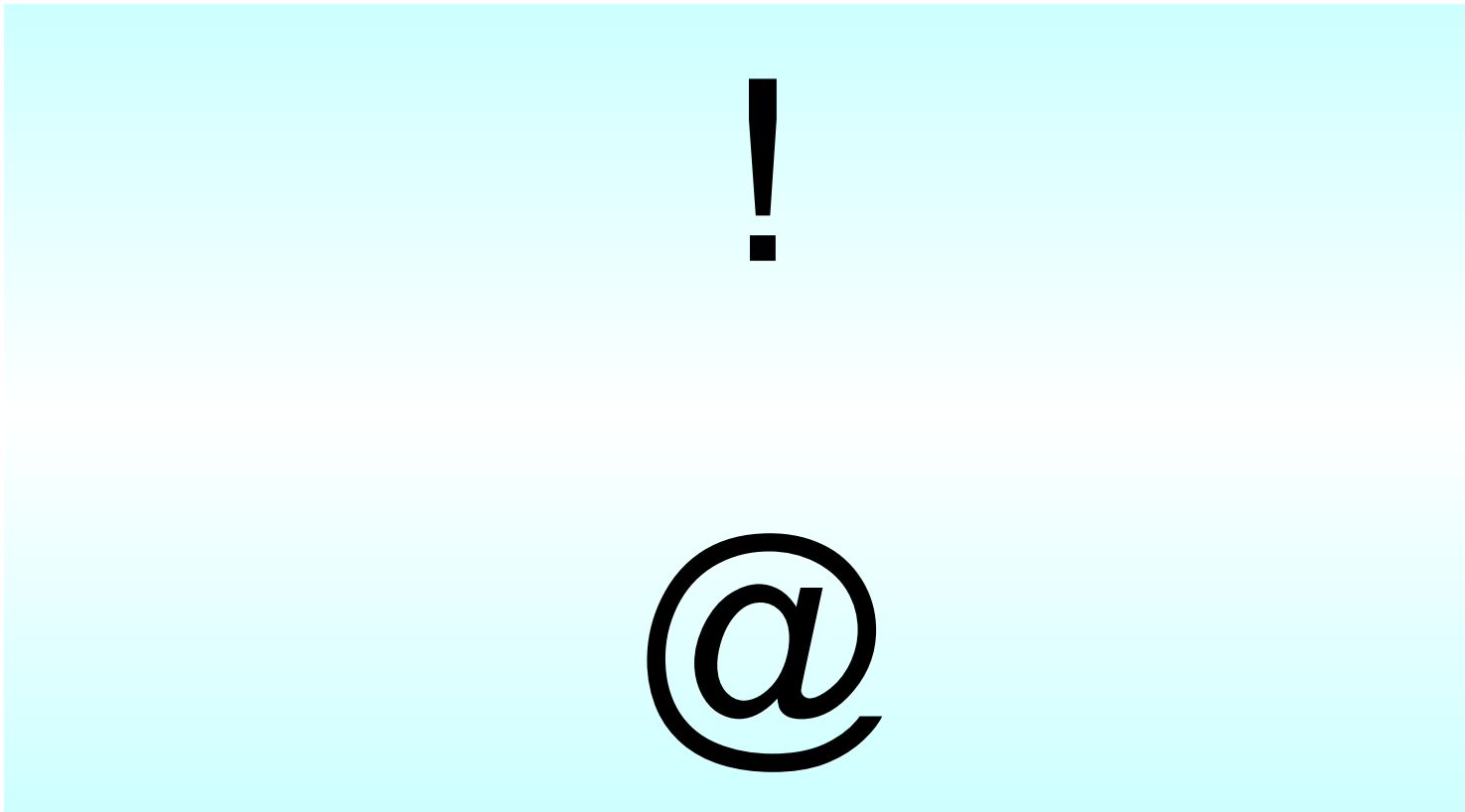
- Initialize Input Mux, Persistence Counters, and Match registers
- Write Match_Parallel Register to set up Data and Mask Registers in Data Comparator
- Write Match_Serial Register to run comparison
- Read Match_Counters to examine the searching hits

Implementation of Search Engine

- Implemented on Altera Stratix II FPGA chip
- Synthesized using Quartus II tools
- Tested on NIOS II Evaluation Board
- Resource Utilization:
  - 27728 ALUT (57%)
  - 24363 total registers
  - 786432 memory bits (31%)
  - 34:41 synthesis time


File Edit View Project Assignments Processing Tools Window Help
ep32_chip
ep32_chip.vhd
crossbar.vhd
search.vhd
persist.vhd
match.vhd
ep32.vhd
Compilation Report - Fl...
Timing Closure Flo...
Compilation Report
Flow Summary
Timing Closure Floorplan
Device: EP2S60F672C5ES
match.vhd
79 SIGNAL
80 SIGNAL
81 SIGNAL
82 SIGNAL
83 SIGNAL
84 SIGNAL
85 SIGNAL
86 SIGNAL
87 SIGNAL
88 SIGNAL
89 SIGNAL
90 SIGNAL
91 SIGNAL
92 SIGNAL
93 SIGNAL
94 SIGNAL
23 entity
24 port
New Quartus II Information
Documentation

Where is the Software?



Concluding Remarks

- When you have your own CPU, everything is easy.
- This search engine is a good example of Hardware-Software Codesign. You can trade off hardware for software and time.

Thank you very much!