

# HDL 이란?



## Hardware Description ?



# HDL 이란?



## ■ Hardware

### ◆ Electronic Circuit

- Analog Circuit
- Digital Circuit

## ■ Description (Why?)

### ◆ Design Verify

- Simulation Model (Describe circuit behavior )
- Netlist for PCB auto-route

## ■ Language (How?)

### ◆ Text (Language)

- EDIF : Netlist Language
- VHDL/Verilog : Simulation Modeling Language

### ◆ Graphic (Schematic)



## Language



- **Syntax**
- **Data types**
  - ◆ **Predefined Types**
  - ◆ **User defined Type**
  - ◆ **Enumerated Types**
- **Objects**
  - ◆ **Scalar**
  - ◆ **Array**
- **Operators and Keywords**



## HDL vs. Programming Language



- | ■ <b>HDL</b>   | ■ <b>Programming Language</b>   |
|--|---|
| ◆ <b>Purpose</b> <ul style="list-style-type: none"><li>➢ Circuits Netlist</li><li>➢ Hardware Simulation</li></ul>    | ◆ <b>Purpose</b> <ul style="list-style-type: none"><li>➢ Executable Code</li></ul>                |
| ◆ <b>Statements behavior</b> <ul style="list-style-type: none"><li>➢ Concurrent</li><li>➢ Sequential</li></ul>       | ◆ <b>Statements behavior</b> <ul style="list-style-type: none"><li>➢ Sequential OLNLY!</li></ul>  |
| ◆ <b>Objects</b> <ul style="list-style-type: none"><li>➢ Signal Name</li><li>➢ Register Implicated</li></ul>         | ◆ <b>Objects</b> <ul style="list-style-type: none"><li>➢ Register Implicated (variable)</li></ul> |
| ◆ <b>Assignment</b> <ul style="list-style-type: none"><li>➢ Wire Connectivity</li><li>➢ Storage assignment</li></ul> | ◆ <b>Assignment</b> <ul style="list-style-type: none"><li>➢ Storage assignment</li></ul>          |



## HDL vs. PL Example



### HDL

```
SIGNAL A,B,C,D,E,clk : bit;  
  
D <= A when sel='0' else B;  
  
regs : process (clk)  
begin  
  if (clk'event and clk='1') then  
    E <= D + C;  
  end if;  
end process;
```

### Programming Language

```
int A,B,C,D,E;  
  
if (sel==0)  
  D = A;  
else  
  D = B;  
  
E = D + C;
```

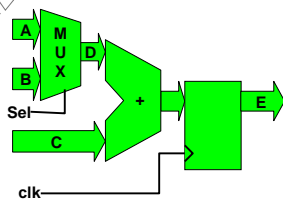


## HDL vs. PL Example (계속)



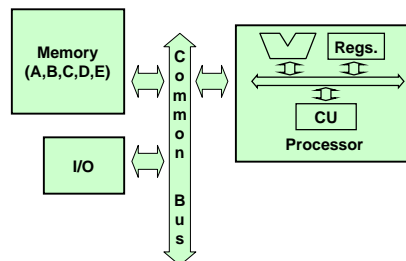
### HDL

- ◆ Instanciated Unit
- ◆ Register Inference



### Programming Language

- ◆ Processor-Memory





## 왜 HDL을 사용하는가?



- “아이디어”를 쉽게 표현 할수 있다
- “설계관리”가 용이하다
- “설계 툴과 공정”에 구애받지 않는다



## HDL은 만능이 아니다

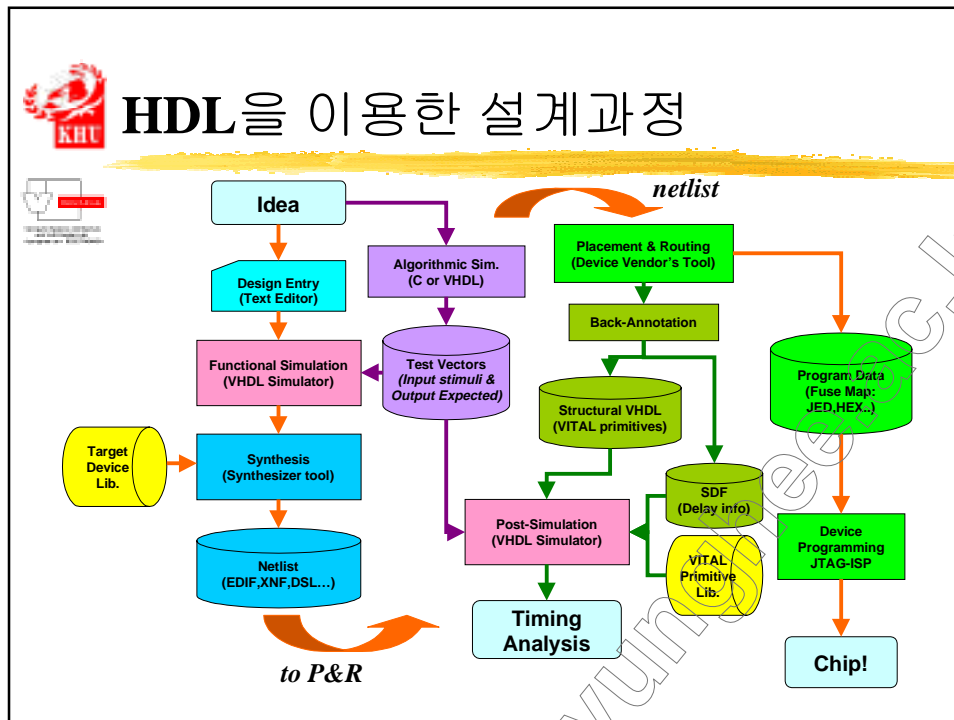


- 합성 가능한 코드와 불가능한 코드

```
process(clk,d)
begin
  if (clk'event and clk='1') then
    zq <= d;
  else
    zq <= 'Z';
  end if;
end process;
```



## HDL을 이용한 설계과정



## VHDL을 시작하려면 필요한 것들...

- Programming Language Compiler
  - ◆ Algorithm Test
  - ◆ Test Vector Generation
- HDL Simulator
  - ◆ Functional Simulation
  - ◆ Timing Simulation
- HDL Synthesizer
  - ◆ HDL to Netlist
  - ◆ Optimization
- Device Vendor Tool (Design Kit)
  - ◆ Place and Route
  - ◆ Timing Extraction



## VHDL Simulator (PC Version)



- **ModelSim (Ex. V-System)**

- ◆ <http://www.model.com>
- ◆ WS/PC Version
- ◆ Support VHDL/Verilog

- **Aldec ActiveVHDL**

- ◆ <http://www.aldec.com>
- ◆ FSM tool included

- **PeakVHDL**

- ◆ <http://www.acc-eda.com>
- ◆ Green Mountain Inc.

- **MyVHDL (서두로직)**

- ◆ <http://www.seodu.co.kr>



## VHDL Synthesizer (General PLD)



- **Metamor Synthesizer**

- **MINC PLSynthesizer**

- **Synopsys FPGA Express**

- **Synplify**

- **Exemplar Galileo/Leonardo**

- **참고:**

- ◆ <http://www.optimagic.com>

# PLD 에 대하여...



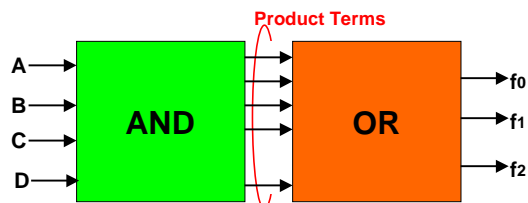
## Programmable Logic Device



# PLD : “Sum of Products”



- Programmable Logic Device
- Combinational Logic Circuits
- ◆ “Sum of Products”
  - AND Plane (product term)
  - OR Plane (sum)





## PLD : ROM and PLA



### ■ Combinational Logic Circuit

#### ◆ ROM Device

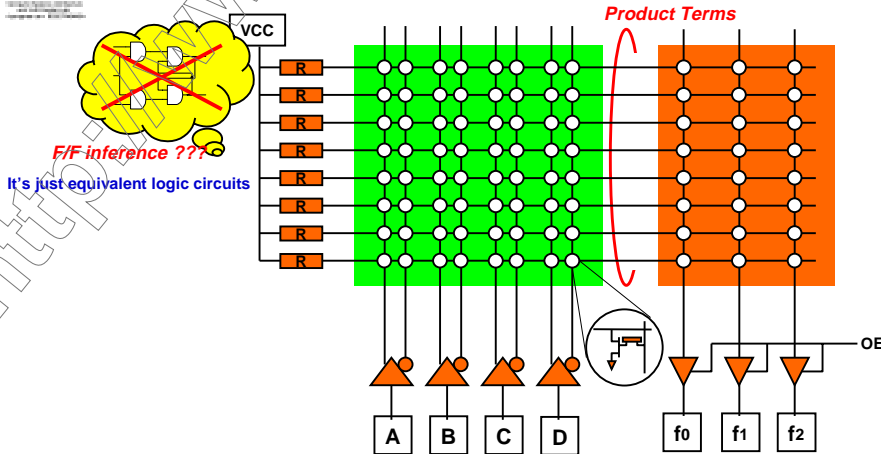
- AND Plane Fixed (Address Decoder)
- OR Plane Programmable

#### ◆ PLA (Programmable Logic Array)

- AND Plane Programmable
- OR Plane Programmable



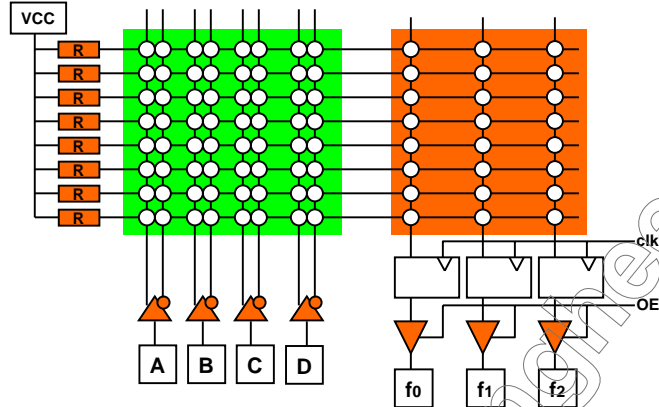
## PLD : PLA Internal



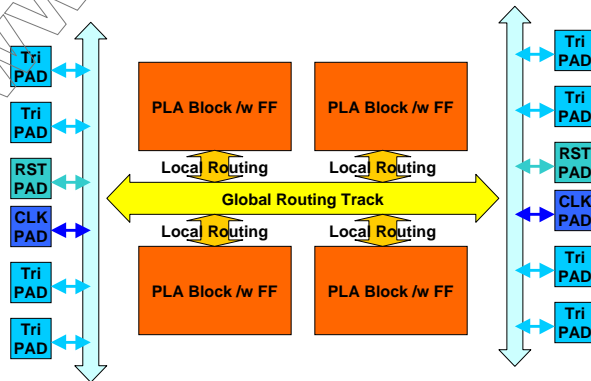




## PLD : PLA with output F/F

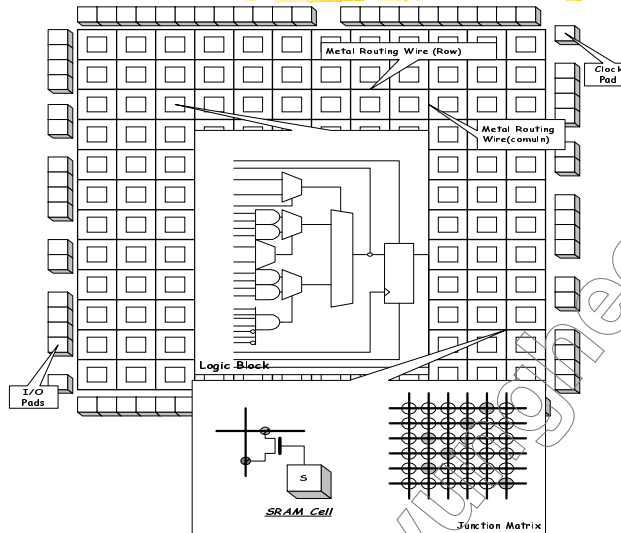


## PLD : Complex PLD

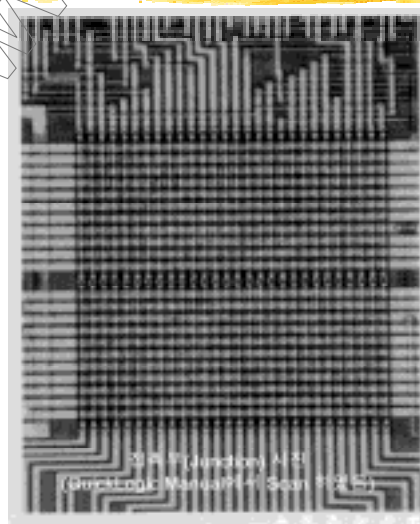




## PLD : FPGA Internal

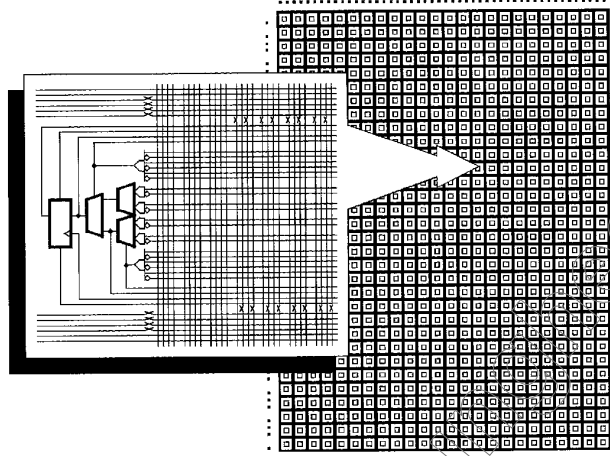


## PLD : Interconnect (Junction Photo)

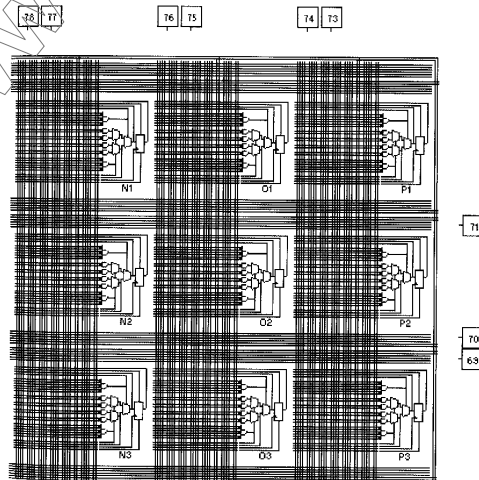




## PLD : Logic Cell (QuickLogic)

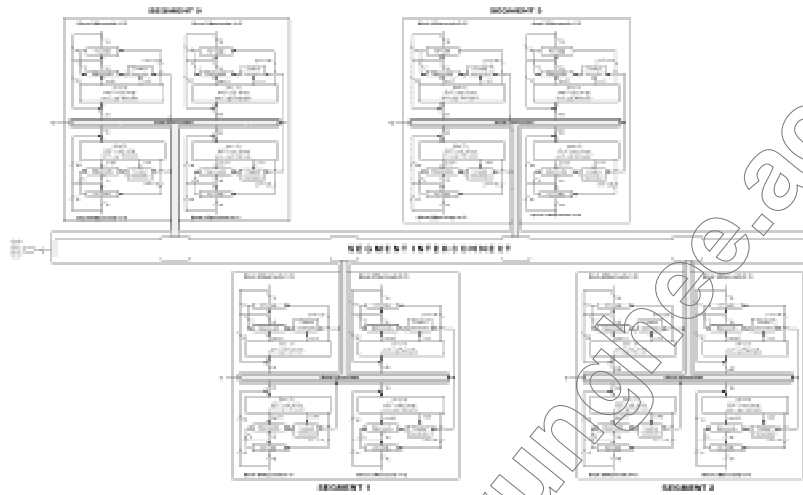


## PLD : Logic Cell (cont'ed)





## PLD : CPLD (Mach5 Internal)



## PLD : CPLD (Mach5 PLA Block)

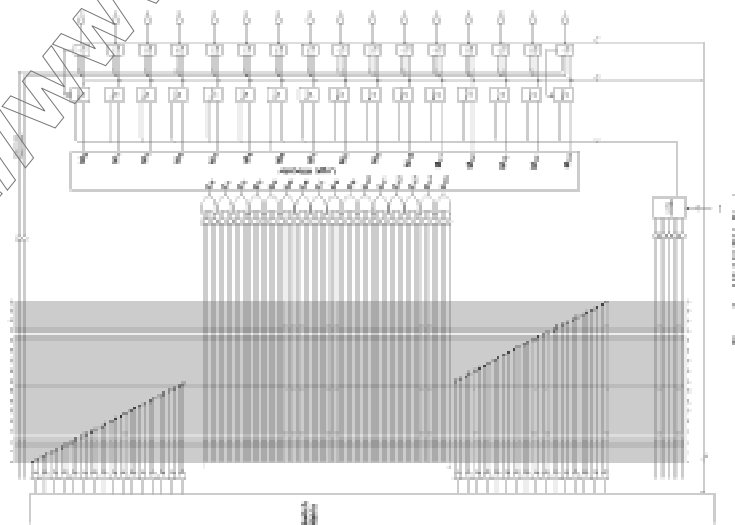


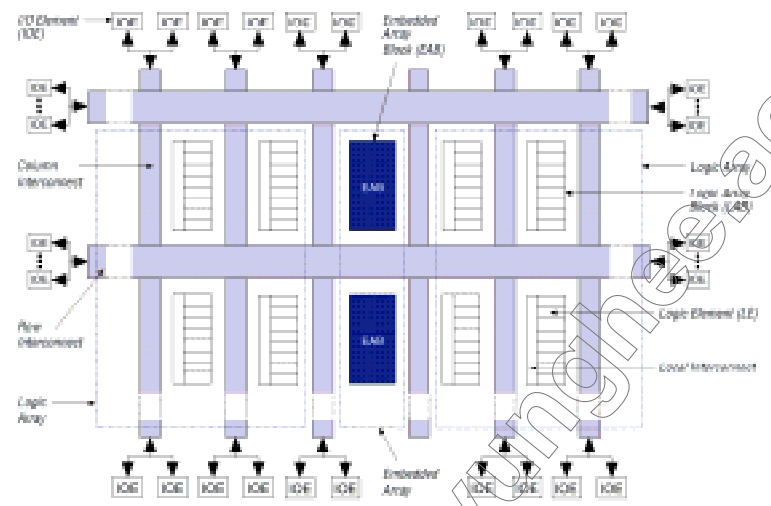
Figure 6. Mach5 PLA Block



# PLD : Flex10k Internal Block



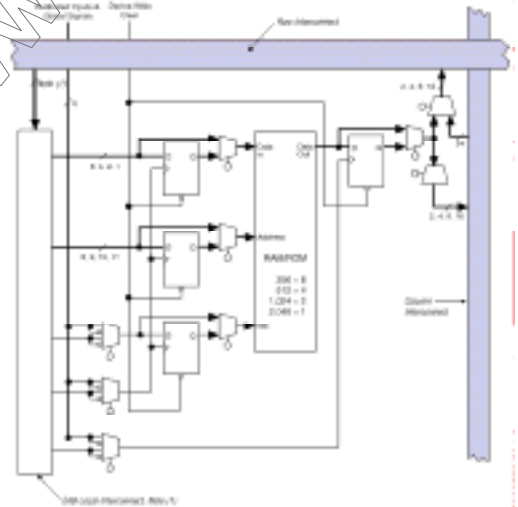
Figure 1. FLEX 10K Device Block Diagram



# PLD : Flex 10k EAB



Figure 2. FLEX 10K Embeddable Array Block

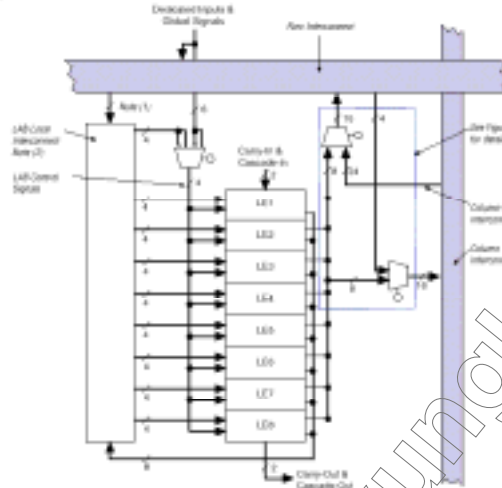




## PLD : Flex 10k LAB



Figure 4-1: PLD Architecture



## PLD : Programming



### ■ ROM

- ◆ Standard Programming Method
- ◆ Generalized Writing Device
  - PROM
  - UV-EPROM
  - EEPROM

### ■ CPLD/FPGA

- ◆ No Standard internal structure
- ◆ Vendor's Writing Device
  - UV-ROM Type
  - PROM Type (OTP)
- ◆ ISP & JTAG
  - EEPROM Type (ISP)
  - SRAM Type



## PLD : Programming - ISP

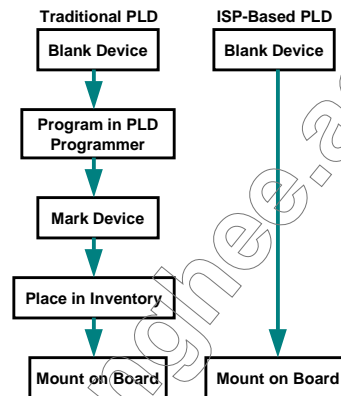


### ■ In-System Programmability

- ◆ Time-to-Market
- ◆ Low price
- ◆ PCB testing capabilities

### ■ JTAG

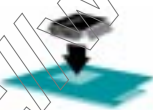
- ◆ Joint Test Action Group
- ◆ Standard IEEE 1149.1-1990
- ◆ PLD Program/Test
- ◆ Verify PCB & PLD Internal
- ◆ Simplified system test



## ISP Benefits



### Mount Unprogrammed



- Eliminates handling of devices
- Prevents bent leads

### Program In-System



- Allows generic end-product inventory
- Specific test protocol or algorithm can be programmed during prototyping, manufacturing or test flow

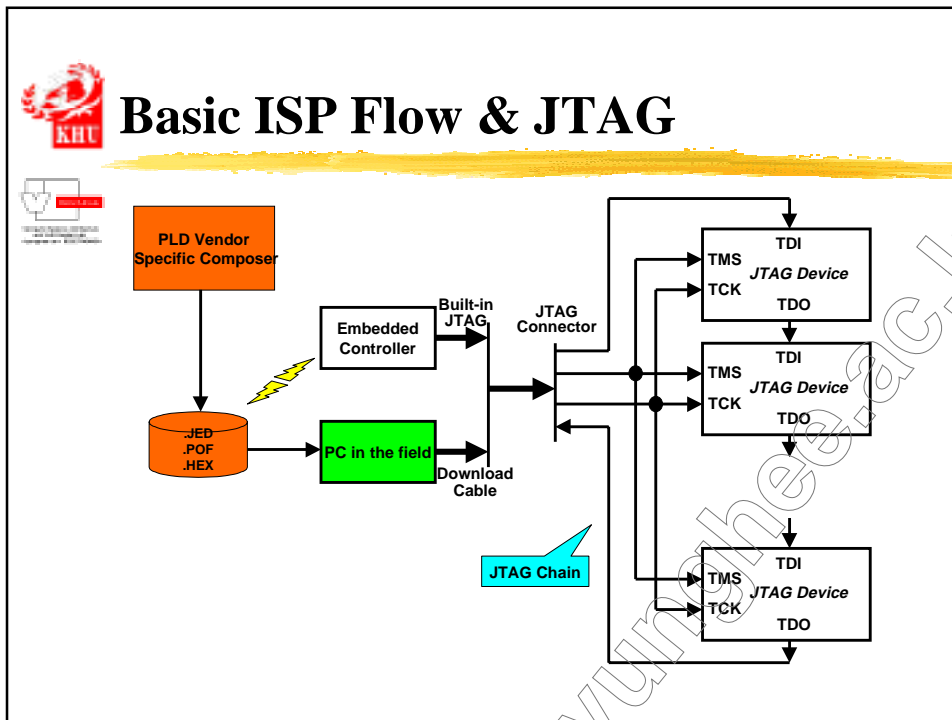
### Reprogram in the Field



- No need to return system for upgrades
- Add enhancements quickly & easily



## Basic ISP Flow & JTAG



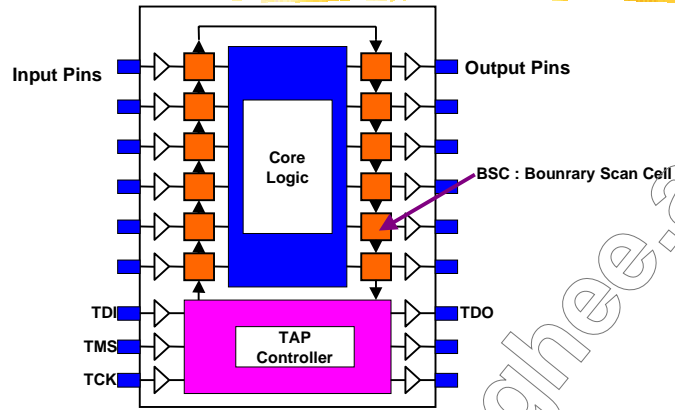
## JTAG - Test Pins

- **TCK - Test Clock**
  - ◆ Provides the clocks (Data Pattern Shifting)
- **TMS - Test Mode Select**
  - ◆ Controls test operation (instruction/data)
- **TDI - Test Data Input**
  - ◆ Receives serial test instruction and data
- **TDO - Test Data Output**
  - ◆ Serial output for test instruction and data
- **/TRST - Test Reset (Optional)**
  - ◆ Resets both the JTAG state machine and the instruction register





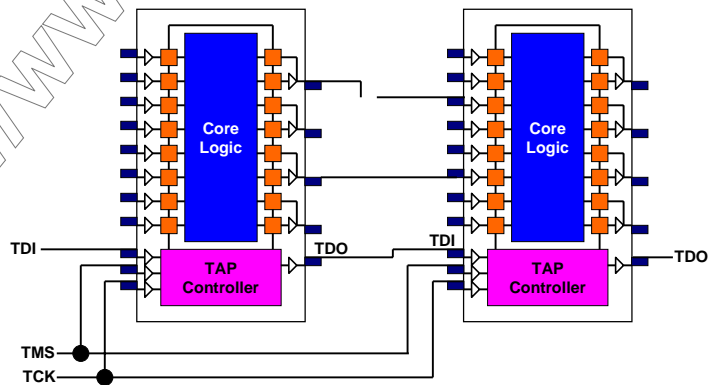
## Boundary Scan Architecture



Test Access Port Controller controls Boundary Scan-Path



## JTAG Connectivity Test Example





## TAP Controller



- **A State Machine to control TAP**
  - ◆ **TAP : TDI,TDO,TMS,TCK,/TRST**
- **Control Operations**
  - ◆ **Shifting Test Data**
  - ◆ **Sampling Data**
  - ◆ **Driving a value**
- **Can pause while shifting registers**
- **Controls input BSC and output BSC separately**

<http://www.csvlsi.kyunghee.ac.kr>