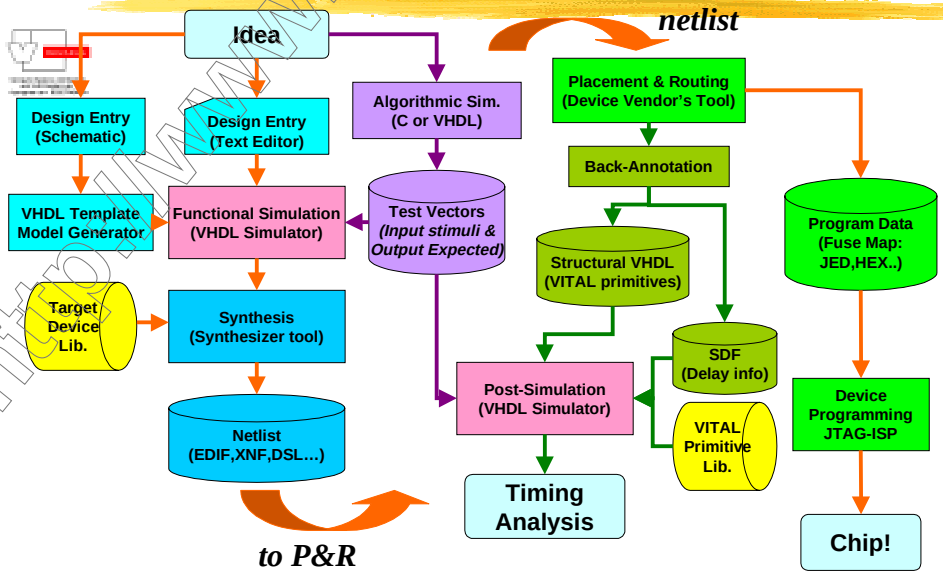
VHDL Design Flow

For Programmable Logic Devices

VHDL Design Flow for PLD

(From Idea to Chip!)



Design Entry

- Text Editor
- Language Sensitive Text Editor
  - Keyword coloring
  - Statement Template
- Schematic Editor
  - Template & Translate (Structural)
- Testbench
  - Converter (Waveformer)

VHDL Functional Simulation

- Library Mapping
  - IEEE
  - STD
  - User Library
- Create Working Library
  - "WORK" Library
- Syntax Analysis & Compile
- Simulation
  - Test Vector : Input Stimuli, Output Expected
  - run testbench

Synthesis

- Convert HDL into Logic
- Translation + Optimization
  - Behavioral, Structural, RTL
  - Component Instanciate
  - Register Inference
  - Logic/Boolean Optimization
  - Area/Speed Optimize Option
  - State machine synthesis
- Technology Mapping
  - Vendor Supplied Lib. (Design Kit)
- Generate Netlist
  - Interface to Placement & Routing Tool : EDIF, XNF, DSL…..

Placement & Route

- Vendor Tool
  - MaxPlus II Compiler, Xact, MachXL. . .
  - Target Device Specific
  - Placement (DataPath, LPM, ROM, RAM)
  - Route (Local/Global)
  - Optimization, Resource Share
  - Fit & Partition
- Back Annotate : Timing Info. Extract
  - Structural VHDL
  - SDF

Post-Simulation

- VITAL
  - VHDL Initiative Toward Asic Library
- Compile Structural VHDL
  - VITAL Primitive Lib. (Device Vendor)
- Use SDF at Simulation Time
  - Delay Info. (Device's Primitive, Routing)
- Timing Analysis : Meet Timing Requirement ?
  - Refine P&R option
  - Refine Synthesis option
  - Algorithm Implementation (Partitioning, Pipeline. . .)

Device Programming

- Junction Info File
  - Fuse map
  - JED, HEX, ...
- Device Programming Method
  - OTP Type Device
  - UV-, EEPROM Type
  - SRAM Type
- Programming Device
  - Dedicated Programmer
  - Universal Programmer
  - Configuration ROM
  - ISP

Design Flow Example

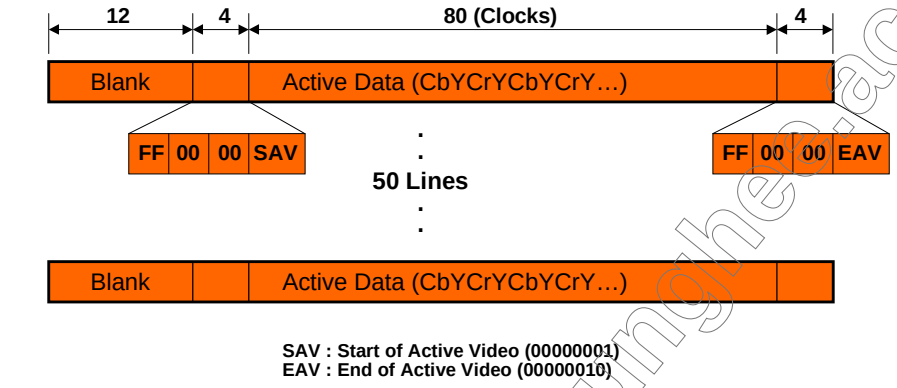
Simplified CCIR Decoder

Tools Used in this Example

- IDE
  - Synario 3.1
- VHDL Simulator
  - ModelTech's V-System
  - Aldec Active VHDL
  - Accolade PeakVHDL
- VHDL Synthesizer
  - Metammor
  - Synplify
  - Exemplar Leonardo
- Placement & Routing (Device Vendor Tool)
  - Vantis Mach XL
  - MaxPlus II
  - QuickWorks

CCIR Decoder

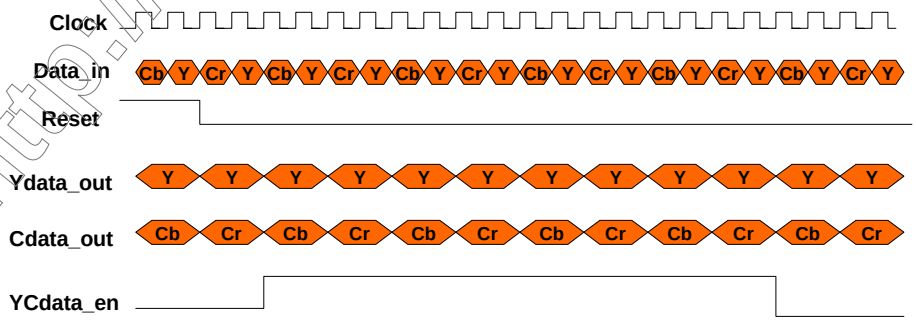
- Decode Digital Video Stream to Equipment
- 8-Bit Digital Video Stream Format



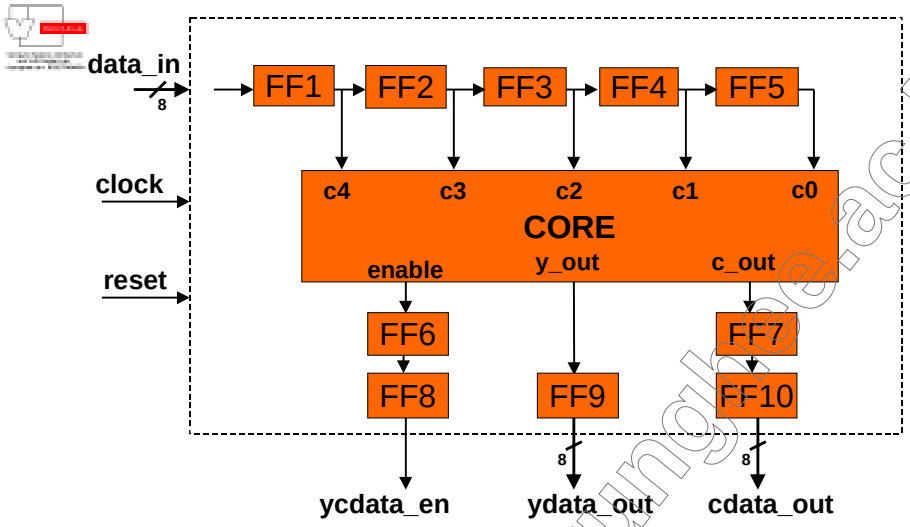
SAV : Start of Active Video (00000001)
EAV : End of Active Video (00000010)

Simplified CCIR Decoder

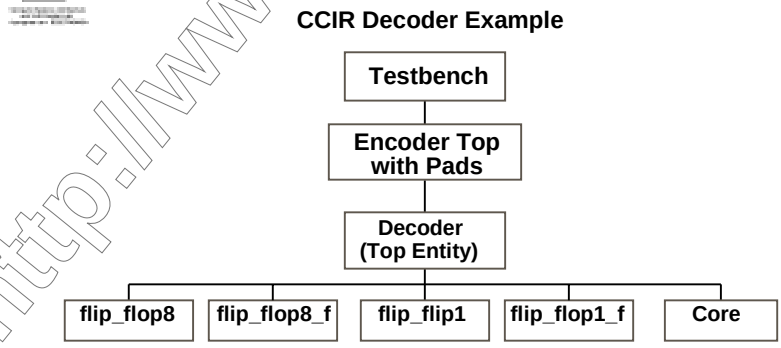
- Decode Active Video Data
  - Input : Active Data, Clock, reset
  - Output : Y, C, Output enable



Design Hierarchy (1)



Design Hierarchy (2)



Design Hierarchy : “CORE”

ENTITY

```
entity core is
  port ( reset  : in std_logic;
         clk    : in std_logic;
         c0     : in std_logic_vector(7 downto 0) ;
         c1     : in std_logic_vector(7 downto 0) ;
         c2     : in std_logic_vector(7 downto 0) ;
         c3     : in std_logic_vector(7 downto 0) ;
         c4     : in std_logic_vector(7 downto 0) ;
         y_out  : out std_logic_vector(7 downto 0);
         c_out  : out std_logic_vector(7 downto 0);
         enable : out std_logic) ;
end core ;
```

Architecture : “CORE”

```
architecture behavioral of core is
begin
   process(clk)
      variable active : boolean := false;  -- Init. NO Effect when Synthesis
      variable get_c  : boolean := true;   -- Use Reset Procedure
      variable count  : integer := 0;
   begin
      if clk'event and clk = '1' then
         if reset='1' then -- Sync. Reset . . . . . .
            count := 0;
            active := false;
            get_c := true;
         else
            if (c0="11111111") and (c1="00000000") and
               (c2="00000000") and  (c3(0)="1") then -- Detect SAV
                 count := 0;
                 active := true;  -- Active Data Start Flag
            else
               if (count = 79) then  -- Active Data Count <80
                  active := false;  -- Active Data End
               end if;
               count := count + 1;
            end if;
```

Architecture : "CORE" (cont'ed)

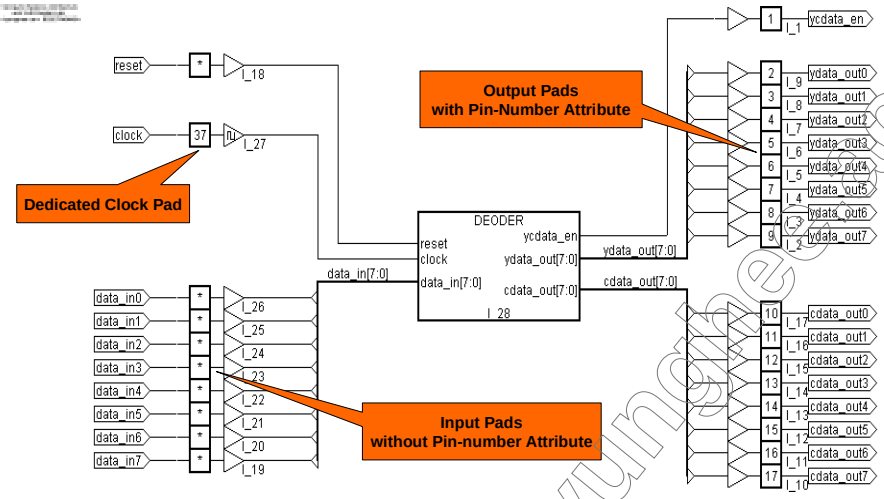
```
            if active then  -- Active Data ?
                enable <= '1';
                if get_c then
                   c_out <= c4;
                   get_c := false; -- Toggle: Next Out
                else
                   y_out <= c4;
                   get_c := true; -- Toggle: Next Out
                end if;
             else
                enable <= '0';
                y_out <= "00010000";
                c_out <= "10000000";
             end if;
          end if;
       end if;
    end process;
end behavioral;
```

Top Module Design

- Structural VHDL
- Attach Pad
  - Some Synthesizer Split Bus
  - Use Text Editor (Buffering by Synthesizer)

    or
  - Use Schematic Editor
- Pin Assignment
  - Pin number attribute on VHDL
  - Pin number attribute on Schematic Editor

Top Module (Schematic Entry)



Top Module (Text Entry)

```
entity decoder_top is
  port ( reset     : in  std_logic;
         clock     : in  std_logic;
         data_in0  : in  std_logic;
         data_in1  : in  std_logic;
         data_in2  : in  std_logic;
         data_in3  : in  std_logic;
         data_in4  : in  std_logic;
         data_in5  : in  std_logic;
         data_in6  : in  std_logic;
         data_in7  : in  std_logic;
         ydata_out0 : out std_logic;
         ydata_out1 : out std_logic;
         ydata_out2 : out std_logic;
         ydata_out3 : out std_logic;
         ydata_out4 : out std_logic;
         ydata_out5 : out std_logic;
         ydata_out6 : out std_logic;
         ydata_out7 : out std_logic;
         cdata_out0 : out std_logic;
         cdata_out1 : out std_logic;
         cdata_out2 : out std_logic;
         cdata_out3 : out std_logic;
         cdata_out4 : out std_logic;
         cdata_out5 : out std_logic;
         cdata_out6 : out std_logic;
         cdata_out7 : out std_logic;
         ycdata_en  : out std_logic ) ;
```

```
attribute pinnum : string;    -- define pinnum attribute
attribute pinnum of clock  : signal is "13";
attribute pinnum of reset  : signal is "8";

attribute pinnum of ydata_out0 : signal is "19";
attribute pinnum of ydata_out1 : signal is "20";
attribute pinnum of ydata_out2 : signal is "21";
attribute pinnum of ydata_out3 : signal is "22";
attribute pinnum of ydata_out4 : signal is "23";
attribute pinnum of ydata_out5 : signal is "24";
attribute pinnum of ydata_out6 : signal is "25";
attribute pinnum of ydata_out7 : signal is "26";

attribute pinnum of cdata_out0 : signal is "31";
attribute pinnum of cdata_out1 : signal is "32";
attribute pinnum of cdata_out2 : signal is "33";
attribute pinnum of cdata_out3 : signal is "34";
attribute pinnum of cdata_out4 : signal is "35";
attribute pinnum of cdata_out5 : signal is "36";
attribute pinnum of cdata_out6 : signal is "37";
attribute pinnum of cdata_out7 : signal is "38";

end decoder_top;
```

Top Module (Text Entry)

```
architecture structural of decoder_top is

 component decoder
  port ( reset    : in  std_logic ;
         clock    : in  std_logic ;
         data_in  : in  std_logic_vector(7 downto 0) ;
         ydata_out : out std_logic_vector(7 downto 0) ;
         cdata_out : out std_logic_vector(7 downto 0) ;
         ycdata_en : out std_logic ) ;
 end component;

 signal data_in   : std_logic_vector(7 downto 0) ;
 signal ydata_out : std_logic_vector(7 downto 0) ;
 signal cdata_out : std_logic_vector(7 downto 0) ;

begin
 data_in <= data_in7 &
            data_in6 &
            data_in5 &
            data_in4 &
            data_in3 &
            data_in2 &
            data_in1 &
            data_in0;

 ydata_out0 <= ydata_out(0);
 ydata_out1 <= ydata_out(1);
 ydata_out2 <= ydata_out(2);
 ydata_out3 <= ydata_out(3);
 ydata_out4 <= ydata_out(4);
 ydata_out5 <= ydata_out(5);
 ydata_out6 <= ydata_out(6);
 ydata_out7 <= ydata_out(7);
 cdata_out0 <= cdata_out(0);
 cdata_out1 <= cdata_out(1);
 cdata_out2 <= cdata_out(2);
 cdata_out3 <= cdata_out(3);
 cdata_out4 <= cdata_out(4);
 cdata_out5 <= cdata_out(5);
 cdata_out6 <= cdata_out(6);
 cdata_out7 <= cdata_out(7);

 udecoder : decoder
        port map ( reset    => reset,
                   clock    => clock,
                   data_in  => data_in,
                   ydata_out => ydata_out,
                   cdata_out => cdata_out,
                   ycdata_en => ycdata_en  );
end structural;
```

Test Vector Generator

- Use Programming Language
- Large Scale Simulation Vector
- Algorithm/Accuracy Test

```
/*********************************
 * Test Vector Generation
 * Project : Decoder
*********************************/

#include <stdio.h>
#include <stdlib.h>
#include <math.h>

typedef unsigned char  byte;

int main(argc,argv)
{
   int i,j;
   int  y_value, cb_value, cr_value;
   byte data[50][100];
   byte y[50][40];
   byte c[50][40];
   byte cb[50][20];
   byte cr[50][20];

   long t_reset;

   FILE *out1;
   FILE *out2;

   out1=fopen("in_vectors","w");
   out2=fopen("out_vectors","w");

   y_value = 0;
   cb_value = 100;
   cr_value = 200;

   for (i=0; i<50; i++)
      for (j=0; j<40; j++)
      {  y[i][j] = y_value;
         y_value++;
         if ( y_value>=10 ) y_value=0;
      }

   for (i=0; i<50; i++)
      for (j=0; j<20; j++)
      {  cb[i][j] = cb_value;
         cr[i][j] = cr_value;
         cb_value++;
         cr_value++;
         if (cb_value>=110)
cb_value=100;
         if (cr_value>=210)
cr_value=200;
      }

for (i=0; i<50; i++)
   for (j=0; j<100; j++)
   {
      if (j>=0 && j<= 11)
         data[i][j] = 0x10;
      else if (j==12)
         data[i][j] = 0xFF;
      else if (j==13)
         data[i][j] = 0x00;
      else if (j==14)
         data[i][j] = 0x00;
      else if (j==15)
         data[i][j] = 0x01;
      else if (j==96)
         data[i][j] = 0xFF;
      else if (j==97)
         data[i][j] = 0x00;
      else if (j==98)
         data[i][j] = 0x00;
      else if (j==99)
         data[i][j] = 0x02;
      else
         if (j%4==0)
            data[i][j] = cb[i][(j-16)/4];
         else if (j%4==1)
            data[i][j] = y[i][(j-17)/2];
         else if (j%4==2)
            data[i][j] = cr[i][(j-18)/4];
         else
            data[i][j] = y[i][(j-17)/2];
   }

   for (i=0; i<50; i++)
      for (j=0; j<100; j++)
         fprintf(out1,"%d\n", data[i][j]);

   for (i=0; i<50; i++)
      for (j=0; j<40; j=j+2)
      {   c[i][j]   = cb[i][j/2];
          c[i][j+1] = cr[i][j/2];
      }

   for (i=0; i<50; i++)
      for (j=0; j<40; j++)
         fprintf(out2,"%3d %3d\n",
                      y[i][j],c[i][j]);

   fclose(out1);
   fclose(out2);

   return 0;
}
```

Testbench

```
entity testbench is end;

architecture behavioral of testbench is
   -- DUT Declare
   component decoder_top
      port(reset        : in std_logic ;
           clock        : in  std_logic;
           data_in0     : in  std_logic;
           . . . . .
           ycdata_en : out std_logic );
   end component;

begin
   -- DUT Instanciate
   inst_top : decoder_top
   port map ( reset      => reset,
              clock      => clock,
              data_in0   => data_in(0),
                  . . . . . .
               ycdata_en  => ycdata_en    );
```

```
-- Reset Stimulus
reset <= '1', '0' after 100 ns;

-- Clock Generator
toggleclock : process(clock)
begin
   clock <= not clock after 20 ns;
end process;
```

Testbench (cont'ed)

```
process(clock)
   -- Input/Output Vector File
   file in_file1 : text is "../test_vector/in_vectors";
   file in_file2 : text is "../test_vector/out_vectors";
    . . . . . .
begin
-------------------------------------------------------------------
-- Falling Edge : Feed Input Vector
-------------------------------------------------------------------
if clock'event and clock='0' then
    . . . . .
    -- Read Input Vector
   readline(in_file1, line_buf1);
   read(line_buf1, line_buf1_field1, good => good_number);
    -- Feed-in to DUT
  data_in <= conv_std_logic_vector(line_buf1_field1, 8);
   . . . . .
end if;
```

Testbench (cont'ed)

```
-------------------------------------------------------------------
-- Rising Edge : Verify Output
-------------------------------------------------------------------
if clock'event and clock='1' then
 if ycdata_en = '1' then  -- Output Enabled ?
     . . . . .
     readline(in_file2, line_buf2);  -- Read EXPECTED Vector
     read(line_buf2, line_buf2_field1, good => good_number);
      . . . . . .                          -- Then Verify DUT output
      if abs(line_buf2_field1 - conv_integer(ydata_out)) /= 0 then
        assert false -- Assert Error Message, if ERROR
        report "ydata = " & to_string(ydata_out) & ". Expected = " &
              to_string(conv_std_logic_vector(line_buf2_field1, 8)) ;
      end if;
     end if;
     . . . . .
```

Functional Simulation 1 (V-System)

- Use Macro File (.do)
  - Compile

```
vlib work
vcom ../design/core.vhd
vcom ../design/flip_flop1.vhd
vcom ../design/flip_flop1_f.vhd
vcom ../design/flip_flop8.vhd
vcom ../design/flip_flop8_f.vhd
vcom ../design/decoder.vhd
vcom ../design/decoder_top.vhd
vcom ../design/testbench.vhd
```

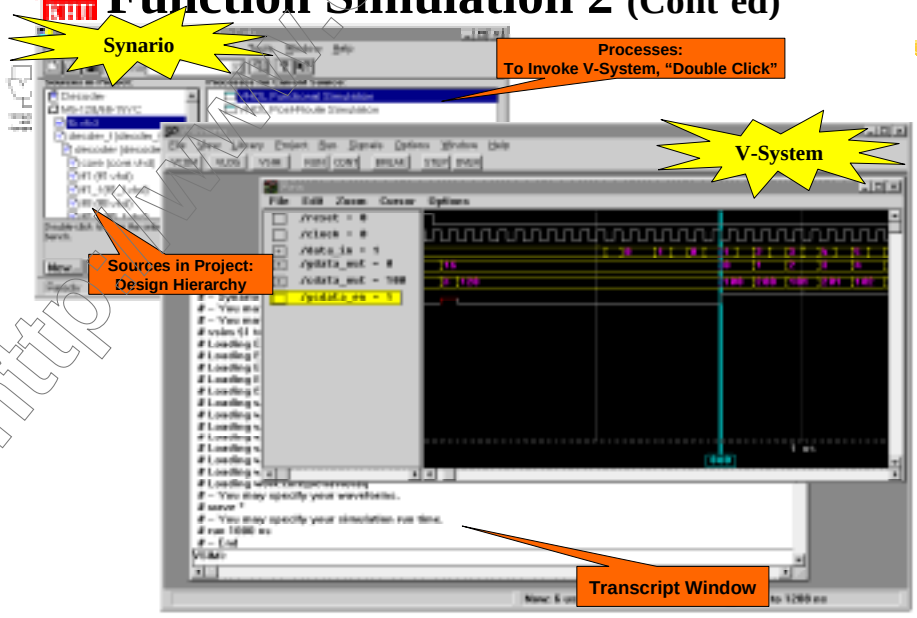
  - Simulate

```
vsim testbench  # Simulate Testbench Entiry
view wave  # Display
wave *
list *
radix decimal
run 20000 ns # Advance Simulator
```

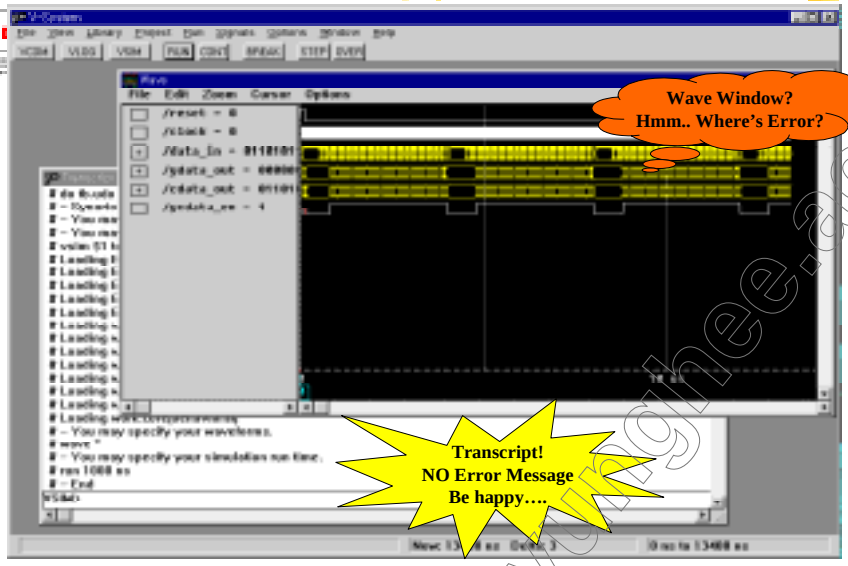
Functional Simulation 2 (Synario)

- New Project
- Import VHDL Source
  - Specify “VHDL Module” or “VHDL Testbench”
  - Synario reads VHDL sources and analysis hierarchy automatically.
  - Entity name and Entity name must be same.
- Function Simulation
  - Select Testbench
  - Double Click “VHDL Function Simulation”
  - Synario create Macro File(.udo) and invoke V-System automatically
- Check Error Report on V-System’s “Transcript” window
- It’s easy! It’s simple!

Function Simulation 2 (Cont’ed)



Function Simulation 2 (Cont'ed)



Synthesis

- Set Option
  - Device Vendor
  - Output (Netlist) Format
  - (Clock) Buffering Insertion
  - Library : IEEE, Synopsys
  - Speed or Area Option
  - Optimization Level
  - Group/Ungroup
  - Flatten/Keep Hierarchy
  - Time Driven
- Set Top Module
- Run Synthesis
- Analysis Result (Log File)
  - Resource Usage
  - Pin Assignment
  - Logic Equation

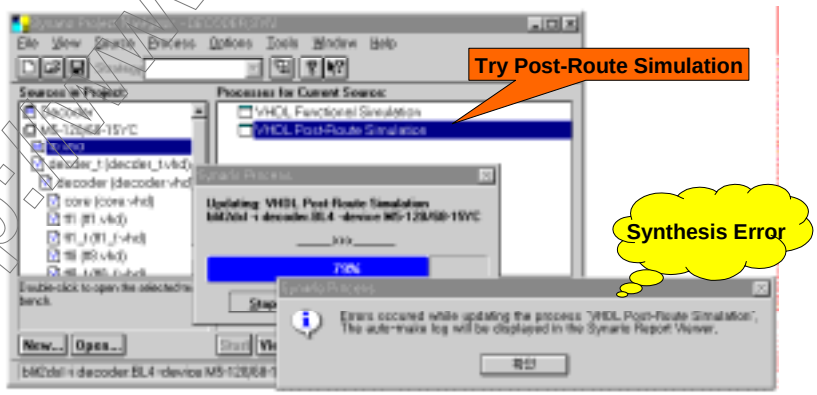
Synthesis 1 (Synario/Metammor/ABEL)

- Choose Device : “MACH 5xx CPLDs”
- Run Synthesis
  - VHDL to ABEL Converter
  - ABEL Optimizer
  - ABEL to DSL Converter
- Synario Synthesis Cause Error!!!
- Check “Automake.log”

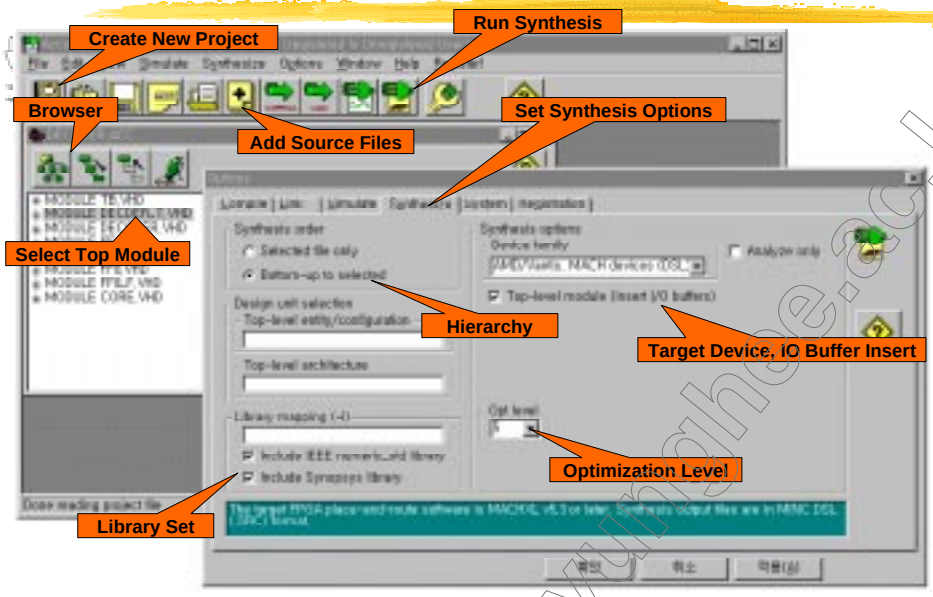
```
Synario Auto-Make Log File
--------------------------
METAMOR VHDL Logic Compiler 3.0.1 (07/03/97)
. . . . .
BLIFOPT  Open-ABEL Optimizer
. . . . .
DIO Flip-Flop Transform (DIOFFT)
. . . . .
BLIF2DSL  Open-ABEL To DSL Translator.
Synario 3.10 Copyright 1993-1997 Data I/O Corp. All Rights Reserved.
      Targeting device:  M5-128/68-15YC.
      Report file:  DECODER.FIT...
Reading Open ABEL 2 (BLIF) file DECODER.BL4...
Writing Constraints File.
      Screening Design...
      Splitting equations to 20 terms...
         Total product terms = 332.
Logical Error 18846: Node U0_n2u contains 34 inputs which exceeds device maximum of 32.
Warning 18822:Design exceeds total macrocells allowed by device.
Warning 18823:Ignoring logical errors that occurred in declarations section.
BLIF2DSL complete - 1 errors, 2 warnings. Time: 2 seconds
Done: failed with exit code: 0002.
```

- Try alternate Synthesizer !!!!!

Synthesis 1 (Cont’ed)



Synthesis 2(PeakFPGA/Metamor)



Synthesis 2(PeakFPGA/Metamor)

- Synthesis VHDL and Generate Netlist
- "Metamor.Log" File
  - Components Structures
  - Infered Flip Flops Structures
  - Optimization Result
  - Device Utilization Report

Synthesis 2(PeakFPGA/Metammor)

```
METAMOR VHDL Logic Compiler 3.0.3 (09/26/97)
Copyright (c) 1992-97 Metamor, Inc. All Rights Reserved
Compiling for: Vantis
analyze....                    3 sec
elaborate design "decoder_top"
                  .....
component : udecoder
component : udecoder/inst_ff
  process :                    0 sec
    Inferred structure :
      flip flop: sr     q 7      FLIP_FLOP8.VHD line 19
      flip flop: sr     q 6      FLIP_FLOP8.VHD line 19
                  ......
elaborate "flip_flop8"           0 sec
component : udecoder/inst_ff2 is same as
        udecoder/inst_ff1
component : udecoder/inst_ff3 is same as
        udecoder/inst_ff1
                  ......
component : udecoder/inst_ff10 is same as
        udecoder/inst_ff9
component : udecoder/inst_core
  process :                    1 sec
    Inferred structure :
      flip flop: sp     get_c 0      CORE.VHD line 29
      flip flop: sr     active 0     CORE.VHD line 29
      flip flop: sr     count 31     CORE.VHD line 29
                  ......
      flip flop: ce     enable       CORE.VHD line 29
      flip flop: ce     c_out 7      CORE.VHD line 29
```

Inferenced Flip Flops

Unused ports

```
                  ......
elaborate "core"                 1 sec
F:\synthesis\design_flow\source\synthesis\CORE.VHD
    line 14 # 703: NOTE: Port 'c3<7>' is not used.
F:\synthesis\design_flow\source\synthesis\CORE.VHD
    line 14 # 703: NOTE: Port 'c3<4>' is not used.
                  ......
elaborate "decoder"              0 sec
elaborate "decoder_top"          1 sec
optimize "decoder_top"....       1 sec
optimize "decoder"....           0 sec
optimize "core"....              8 sec
optimize "flip_flop8_f"....      0 sec
optimize "flip_flop1_f"....      0 sec
optimize "flip_flop1"....        0 sec
optimize "flip_flop8"....        0 sec
  flip flops with synchronous reset = 99
  flip flops with synchronous preset = 1
  flip flops with no set or reset, clock enable = 17
  combinational logic area estimate = 191 two input gates
format....                       1 sec
7 Notes (see log file).
no errors.
compile time = 15 sec
peak dynamic memory allocation = 7.001 Mbyte.
```

Final Report

Place & Route (MachXL)

- **Vendor Tool**
  - **Vantis Device : MachXL 5.3 or later**
  - **Implement “Netlist” on the Chip**
  - **Place Digital Primitives**
  - **Interconnect (Route) Placed Primitives**
- **P&R : Create PLD Fusemap File (.JED)**
  - **Open Netlist File**
  - **Device -> Parameters -> Select Target Device**
  - **Projects -> Build All**
- **Back Annotation : Extract Timing Info**
  - **Setting -> Options -> Timing Model -> VHDL**
  - **Projects -> Generate Timing Model**
  - **Projects -> Generate Timing Report**

Place & Route (MaxPlus II)

- Vendor Tool
  - Altera Device : MaxPlus II
  - Interconnect (Route) Placed Primitives
- P&R, Fitting and Extract Timing Info
  - Open Netlist File and Set Project
    - File -> Open (.EDF) -> Project -> Set Project to . . . . .
  - Compiler :Select Device
    - Assign -> Device *(select device)*
  - Compiler : Netlist(EDIF) File Reader Option
    - VHDL Synthesizer : Synopsys/Exemplar/ViewLogic/Synplify. . .
    - Compiler Menu -> Interface -> EDIF Netlist Reader Setting
  - Compiler : Timing Info Extract Option
    - VHDL-87/93, SDF Version, VITAL
    - Compiler Menu -> Interface -> VHDL Netlist Writer Setting
  - Compiler : Timing Info Extract Set
    - Compiler Menu -> Interface -> VHDL Netlist Writer *(Check)*
  - Compiler : "Start"

Place & Route Report (MachXL)

```
R MACHXL3.9.3.22 - (c) Copyright MINC
        Incorporated 1987-1997

 MODULES :
 ..... Tool Info
 SWITCH VALUES :
 ..... Used Option Switch Info

EQUATIONS FOR SYSTEM

INPUT SIGNALS (10) :
 ..... I/O Pin Info
OUTPUT SIGNALS (17) :
 ..... I/O Pin Info

PHYSICAL NODE SIGNALS (105) :
     decoder.udecoder.core.inst_core.n4r
     decoder.udecoder.core.inst_core.n4s
     decoder.udecoder.core.inst_core.n4t
......
REDUCED EQUATIONS:
decoder.udecoder.core.inst_core.n4r.EQN =

     decoder.udecoder.flip_flop8.inst_ff2.n0h*

     /decoder.udecoder.flip_flop8.inst_ff3.n0a*
......
```

```
PARTITIONING SOLUTIONS :
FUSEMAP FILES FOR SOLUTION 1:
DEVICE SELECTION:
..... Device Info
    Device      Man  Fam Pack Temp  ICC    TPD    Fmax    Price  User

==> M5-128/68-15YC   AMD CMOS QFP  COM   138.0ma 18.0ns  55.0MHz
       $ 26.92   0 0
16) M5-128/68-10VI   AMD CMOS TQFP EXT   188.0ma 11.5ns  87.0MHz $
       53.12   0 0
..... Pinout Info
WIRELIST  Sat Oct 11 11:07:37 1997
```

Signal	Device	Pin
reset	MV128_68_1	7
clock	MV128_68_1	13
data_in0	MV128_68_1	58
data_in1	MV128_68_1	55
data_in2	MV128_68_1	26
. . . . .	. . . . .	
cdata_out3	MV128_68_1	84
cdata_out2	MV128_68_1	38
cdata_out1	MV128_68_1	19
cdata_out0	MV128_68_1	62
ycdata_en	MV128_68_1	20

Place & Route Report (MaxPlus II)

- Device Utilization
- LC (Logic Cell) Usage
- IO Pin Usage
- Pin Assign

Post Simulation

- Use same testbench and Test Vector
- Check Functionality and Delay
- Simulation with “Back Annotated” Timing Model
- Use Vendor Supplied VITAL/Timing Model Library
  - Library Name : “MINC_VHD” (Vantis), “ALT_VTL” (Altera)
- Use Macro Script File (.do)
- Manage Directory
  - Design Entry
  - Test Vectors
  - Functional/Post Simulation
  - Synthesis
  - Timing Model Library

Manage Directory

- Decoder — Design Top
  - Design — Design Entry (.vhd,...)
  - Functional — Functional Simulation (.do)
  - Timing — Timing Simulation (.do)
  - Synthesis — Synthesis Working Dir (.bat) Synthesizer Project Files
    - model — Generated Timing Model (.VHD,.SDF)
  - test_vector — Test Vector and C Source, Makefile
  - mach_vital_src — Timing library Source (.vhd)

Post Simulation (V-System/Mach)

- Use Macro File
  - Compile

```
# Generic Timing Simulation
#-----------------------------------------------------
vlib minc_vhd
vmap minc_vhd minc_vhd
vcom -87 -work minc_vhd ../mach_vital_src/postmods.vhd
# Timing Simulation
#-----------------------------------------------------
vlib work
vcom -87 ../synthesis/model/decoder_top.vhd
vcom -87 ../design/testbench.vhd
```

  - Simulation

```
vsim -sdftyp inst_top=../synthesis/model/decoder_top.sdf
testbench
view wave
wave *
list *
radix decimal
run 5000
```

Post Simulation (V-System/MaxPlus II)

- Use Macro File
  - Compile

```
vlib alt_vtl
vmap alt_vtl alt_vtl
vcom -87 -work alt_vtl ../altera_vital_src/alt_vtl.vhd
vcom -87 -work alt_vtl ../altera_vital_src/alt_vtl.cmp

vlib work
vcom -87 ../synthesis/decoder_top.vho
vcom ../design/testbench.vhd
```

  - Simulation

```
vsim -sdftyp inst_top=../synthesis/decoder_top.sdo testbench
view wave
wave *
list *
radix decimal
run 5000
```


Post Simulation (V-System)

- Check Functionality and Violation (Message Transcript Window)
- Check Timing Delay