# FLUXO DE DESENVOLVIMENTO - FPGA

## WorkFlow FPGA

22 de setembro de 2015

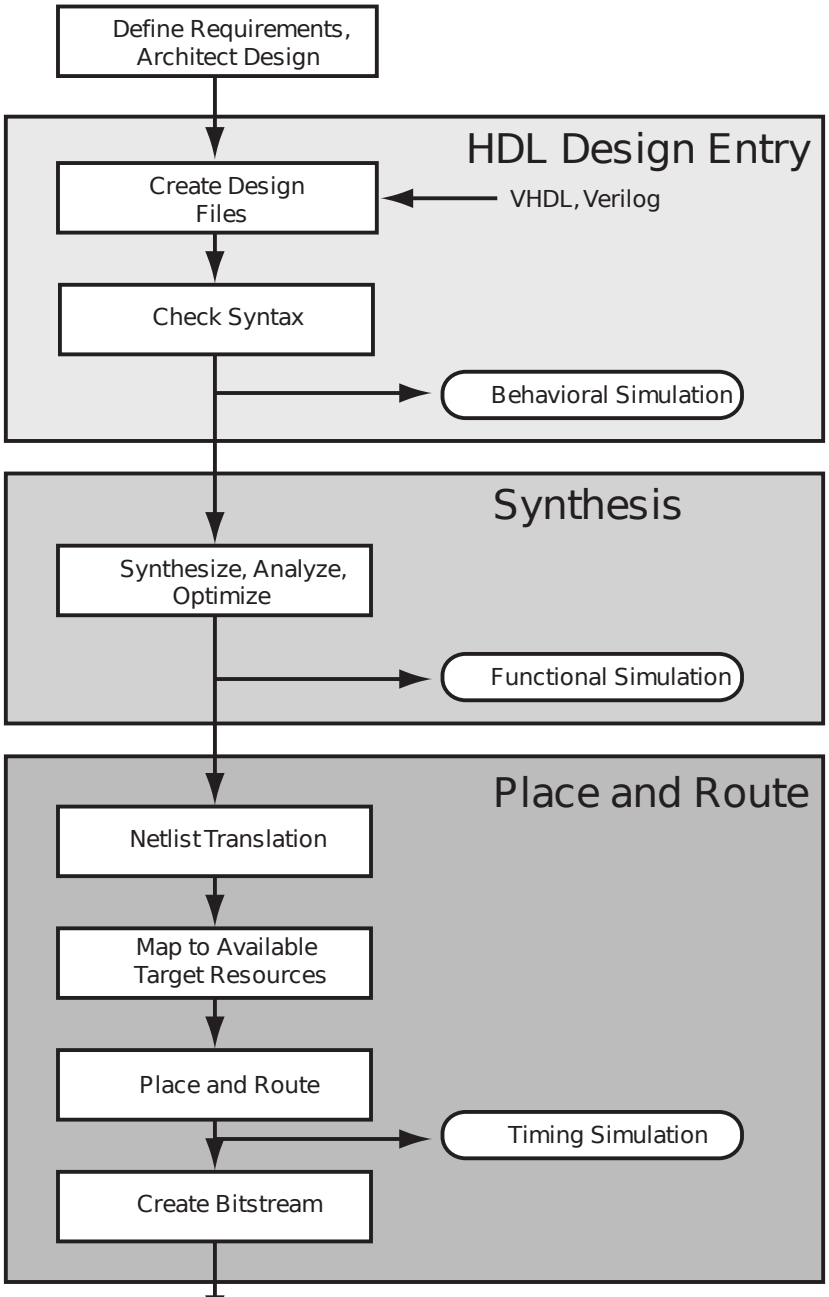
Rafael Corsi Ferrão - IMT

rafael.corsi@maua.br
http://www.maua.br

# CONTEÚDO

# CONTEÚDO

# FLUXO DE DESENVOLVIMENTO

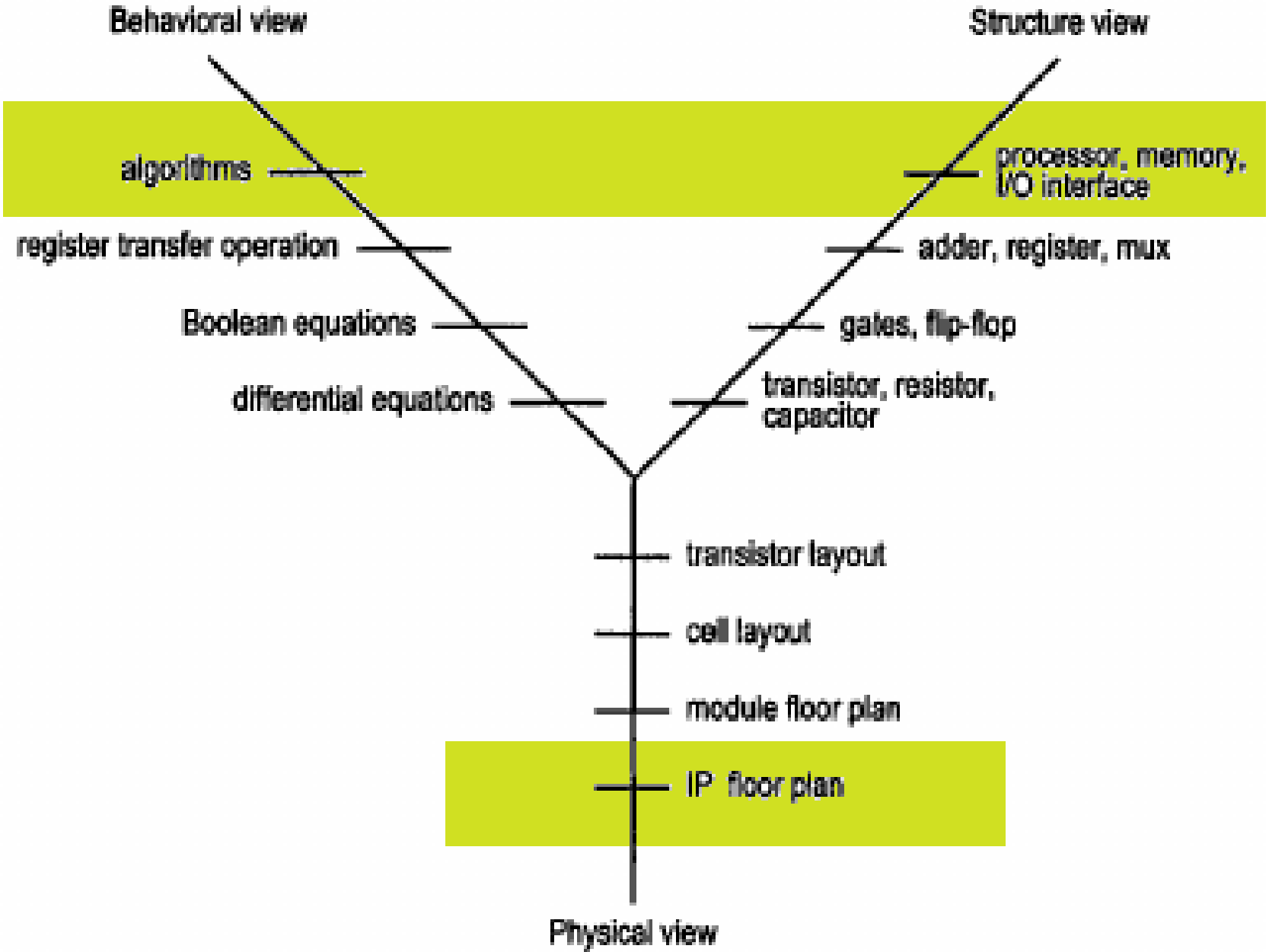


# REQUISITOS E ARQUITETURA

Define o escopo do projeto, as tecnologias utilizadas e a solução proposta

É a etapa mais importante de um projeto digital, onde é definido a solução adotada para o problema (em um nível hierárquico alto), seus diagramas de blocos, especificações e modos de operação.

# NÍVEIS DE ABSTRAÇÃO



## Definição

É o processo de interpretação do HDL em elementos lógicos e implementáveis.

Diferentes abordagens podem ser tomadas para a mesma descrição, portanto é importante a verificação funcional do projeto nessa etapa. É aqui também verifica-se por erros de sintaxe.
O interpretador irá mapear a lógica descrita no HDL em registradores, máquinas de estado, ALUs . . . . Será associado ao arquivo gerado (netlist) as características de tempo dos componentes.

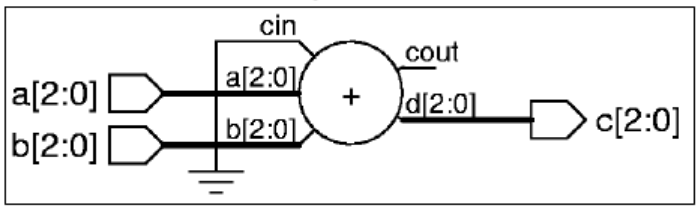
# SÍNTESE

descrição VHDL

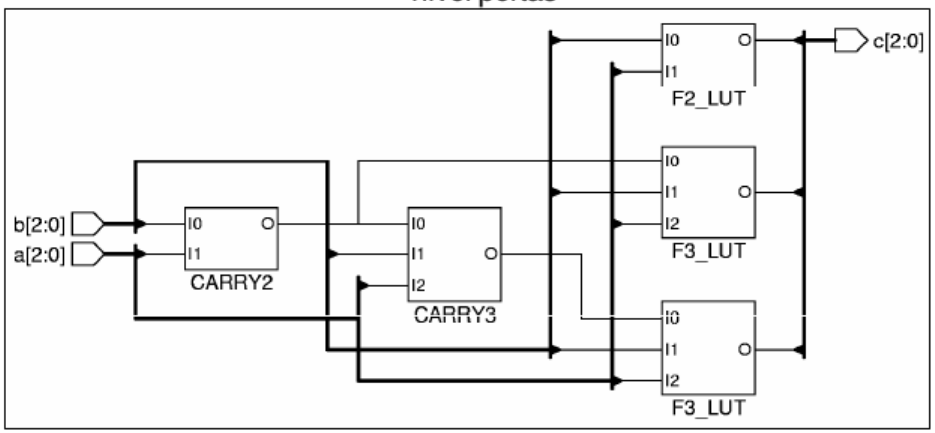
```
ENTITY soma IS
  PORT (a, b : IN  INTEGER RANGE 7 DOWNTO 0;
        c    : OUT INTEGER RANGE 7 DOWNTO 0);
END soma;

ARCHITECTURE teste OF soma IS
BEGIN
  c <= a+b;
END teste;
```

nível RTL




nível portas




otimização velocidade / área

# INSTANCIAÇÃO VS INFERÊNCIAÇÃO

## Instanciação

Instanciação é o processo de implementar funcionalidades pré definidas, é uma escolha direta do desenvolvedor

Torna o design mais otimizado porém menos portátil (entre famílias e fabricantes)

## Inferênciação

Inferênciação é o processo de síntese que com base no HDL aloca funcionalidades pré definidas, não é uma escolha direta do desenvolvedor

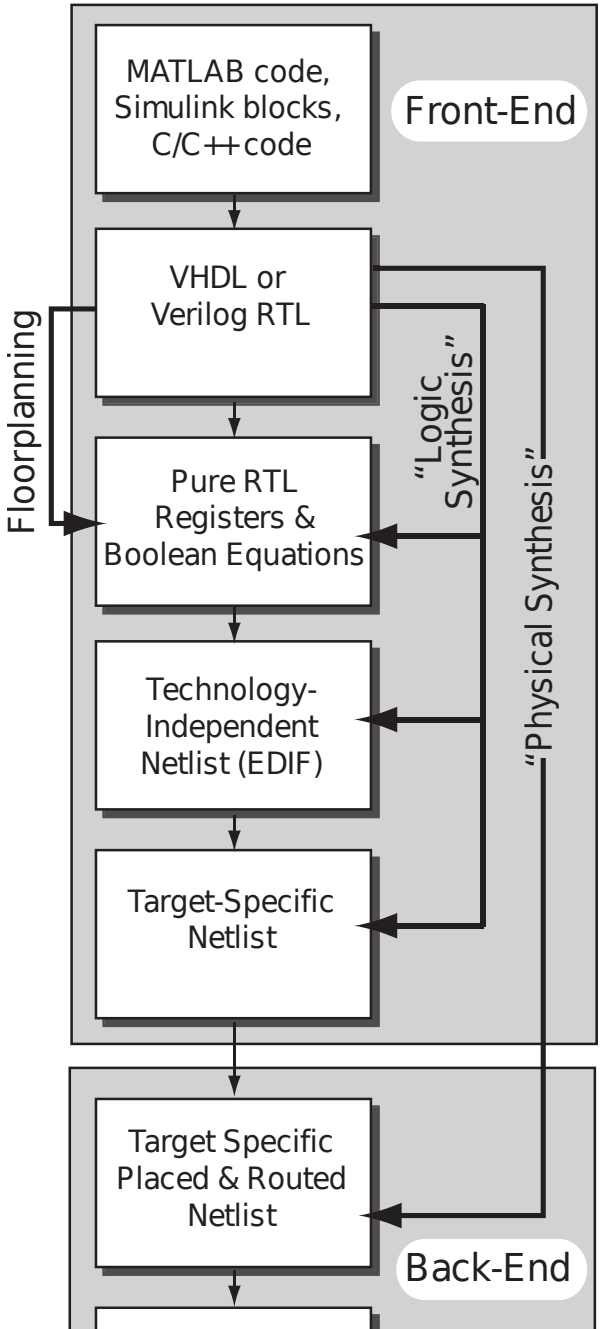
Torna o design mais portátil, já que permite que a ferramenta instancie as funcionalidades.

# OTIMIZAÇÕES

As seguintes otimizações podem ser escolhidas no processo de sintetização:

- Área: otimiza-se o design para a utilização da menor quantidade de elementos lógicos, tornando a frequência máxima de operação menor.
- Tempo : Otimiza-se o design para uma maior frequência, o que torna o design maior em termos de células lógicas
- Potência : O design é otimizado para o menor consumo energético

# SÍNTESE -> MATLAB



## Definição

É o processo que decide onde será alocado as portas lógicas e registradores interpretadas pela etapa de síntese.

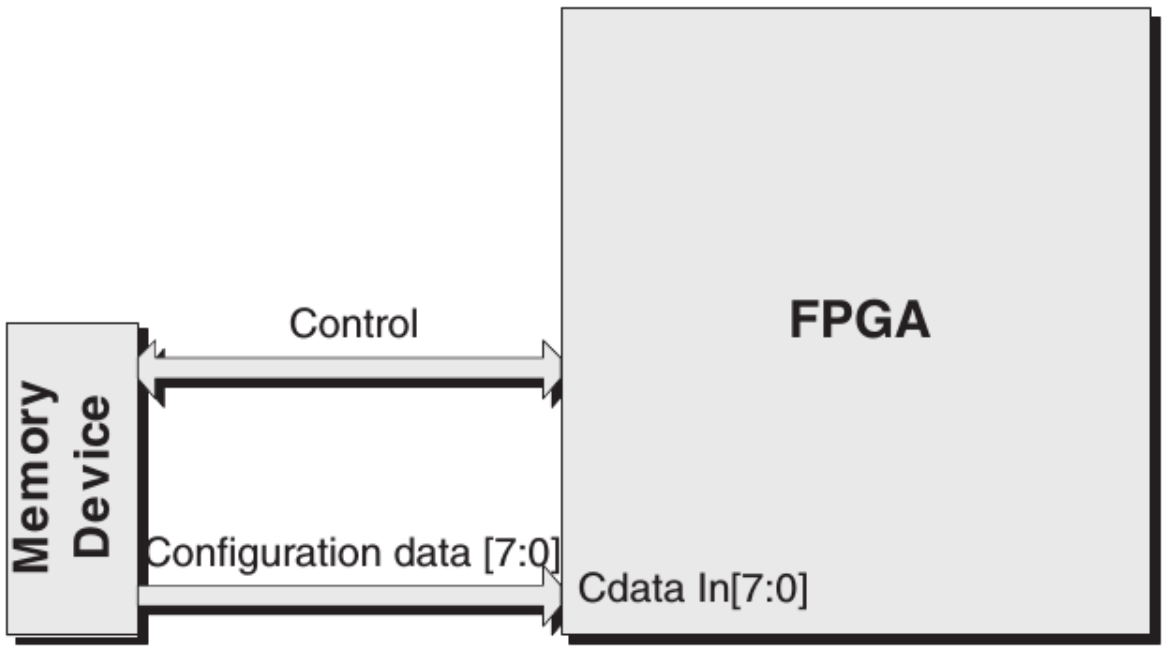
Etapa importante do fluxo de desenvolvimento onde a ferramenta decidirá pelo melhor local para mapear as lógicas, problemas referente a propagação de sinais são levados em conta.
Essa etapa gera um arquivo chamado de Netlist

# CONTEÚDO

A forma mais comum para gravação de FPGAs é pelo o padrão Joint Test Action Group (JTAG). Duas formas de gravação são utilizadas:

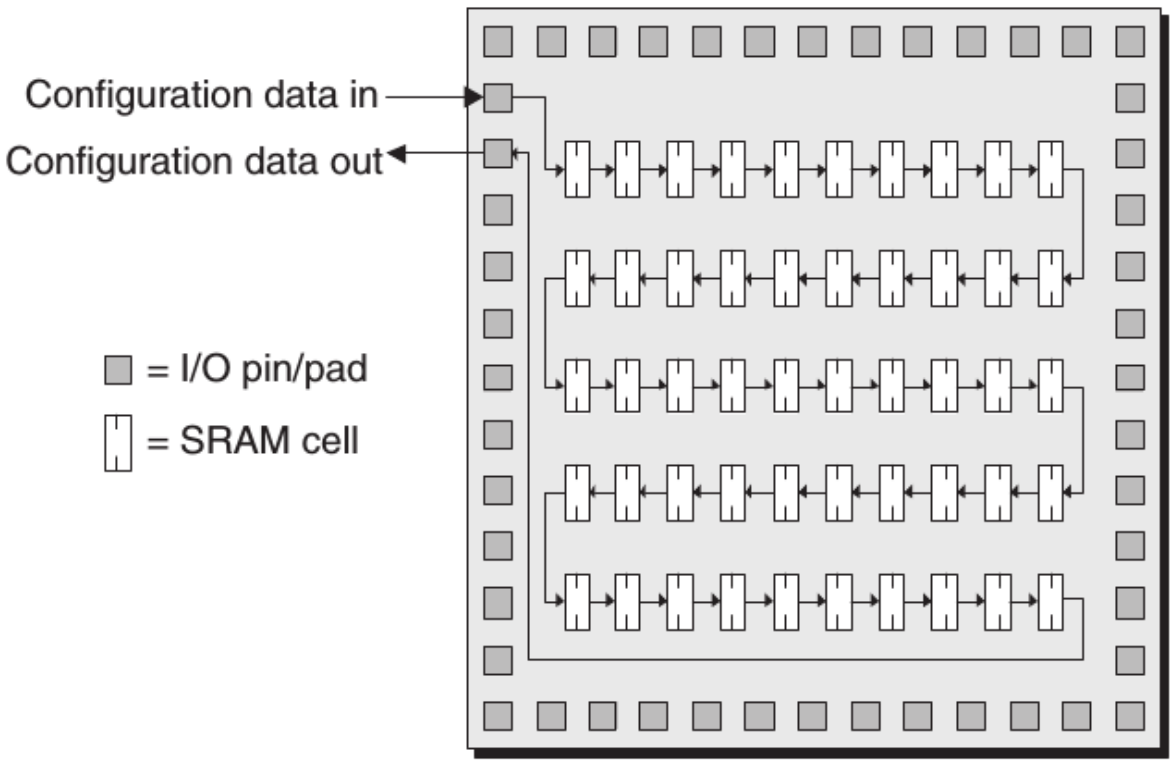
- FPGA : Grava-se diretamente a FPGA (SRAM), processo rápido porém perde-se a gravação com a ausência de energia elétrica
- Memória Flash : Grava-se o Netlist na memória que é carregado na FPGA durante sua inicialização

# INICIALIZAÇÃO - MEMÓRIA



# SRAM FPGA

FPGAs do tipo SRAM podem ser visualizadas como um grande shit-register:



# CONTEÚDO

# XILINX VIVADO

- Software usado para a programação e desenvolvimento em FPGAs Xilinx
  - `http://www.xilinx.com/products/design-tools/vivado/index.htm`
- funciona só para novas famílias de FPGA Xilinx (7 >=)
- possui ambiente de simulação integrado
- versão gratuita (WebPACK)
- Linux e Windows

# TUTORIAIS

A Xilinx disponibiliza vários tutoriais para sua ferramenta :

- `http://www.xilinx.com/training/vivado/index.htm`

# TIPOS DE ARQUIVO

- .vhd : Arquivos VHDL
- .XDC (Xilinx Design Constrains) : Arquivos de configuração da FPGA e da ferramenta, possui os mapeamentos paras os pinos, tipos de sinais, e configurações de clk, ....
- SDC (Synopsys Design Constrains) : Similar ao .xdc

# XDC

```
## Clock signal
#Bank = 35, Pin name = IO_L12P_T1_MRCC_35,                    Sch name = CLK100MHZ
set_property PACKAGE_PIN E3 [get_ports CLK]
    set_property IOSTANDARD LVCMOS33 [get_ports CLK]
    create_clock -add -name sys_clk_pin -period 10.00 -waveform {0 5} [get_ports CLK]

## Switches
#Bank = 34, Pin name = IO_L21P_T3_DQS_34,                     Sch name = SW0
set_property PACKAGE_PIN U9 [get_ports {SW[0]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {SW[0]}]
#Bank = 34, Pin name = IO_25_34,                              Sch name = SW1
set_property PACKAGE_PIN U8 [get_ports {SW[1]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {SW[1]}]
#Bank = 34, Pin name = IO_L23P_T3_34,                         Sch name = SW2
set_property PACKAGE_PIN R7 [get_ports {SW[2]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {SW[2]}]
#Bank = 34, Pin name = IO_L19P_T3_34,                         Sch name = SW3
set_property PACKAGE_PIN R6 [get_ports {SW[3]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {SW[3]}]
#Bank = 34, Pin name = IO_L19N_T3_VREF_34,                    Sch name = SW4
set_property PACKAGE_PIN R5 [get_ports {SW[4]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {SW[4]}]
#Bank = 34, Pin name = IO_L20P_T3_34,                         Sch name = SW5
set_property PACKAGE_PIN V7 [get_ports {SW[5]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {SW[5]}]
#Bank = 34, Pin name = IO_L20N_T3_34,                         Sch name = SW6
set_property PACKAGE_PIN V6 [get_ports {SW[6]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {SW[6]}]
```