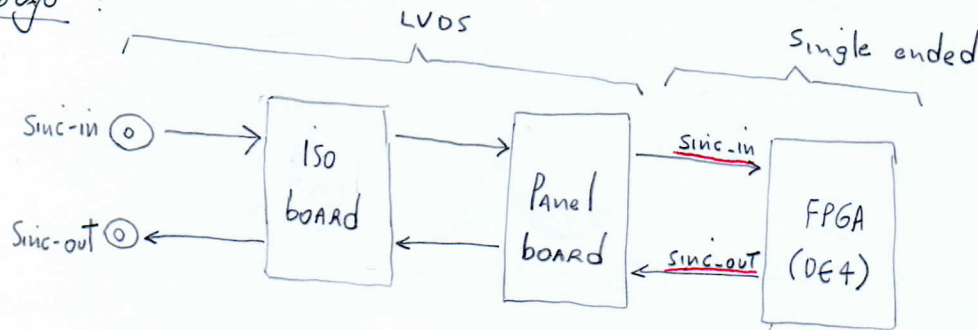
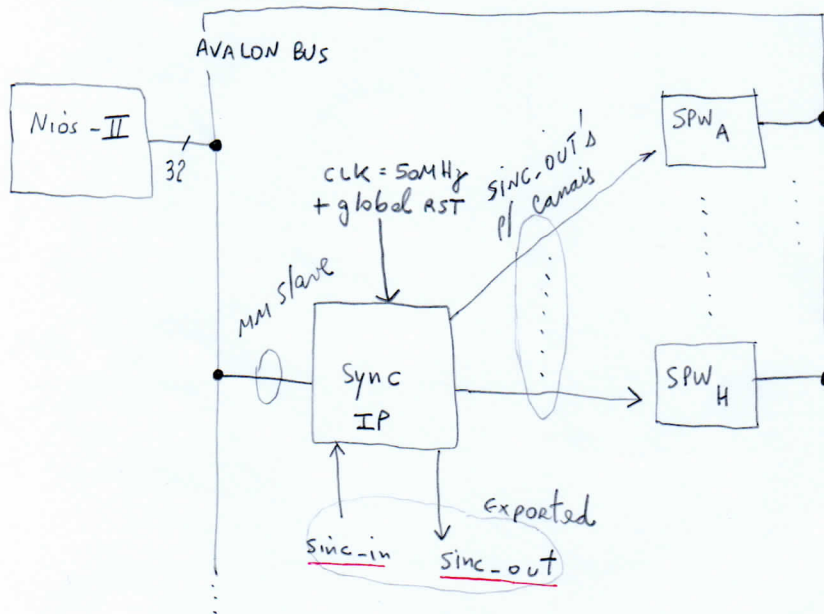


Concepo :

A). HW :

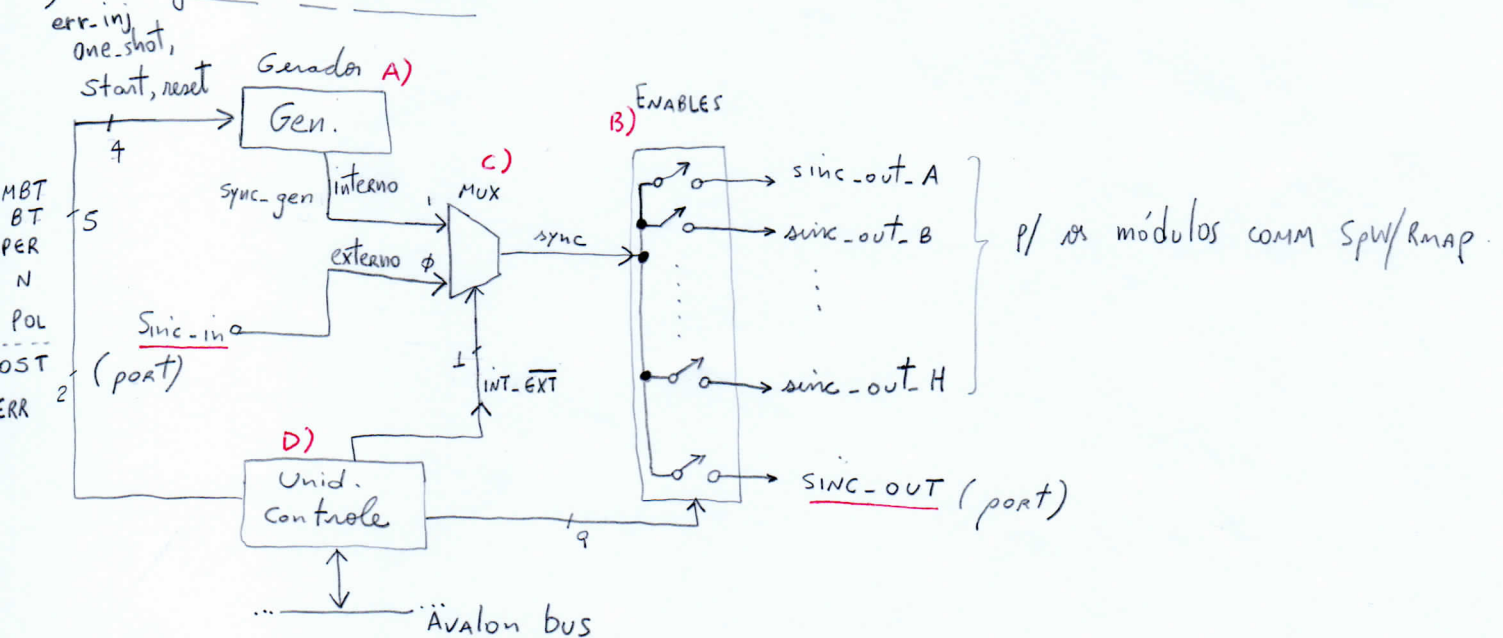


B). Platform Designer (Sistema) :



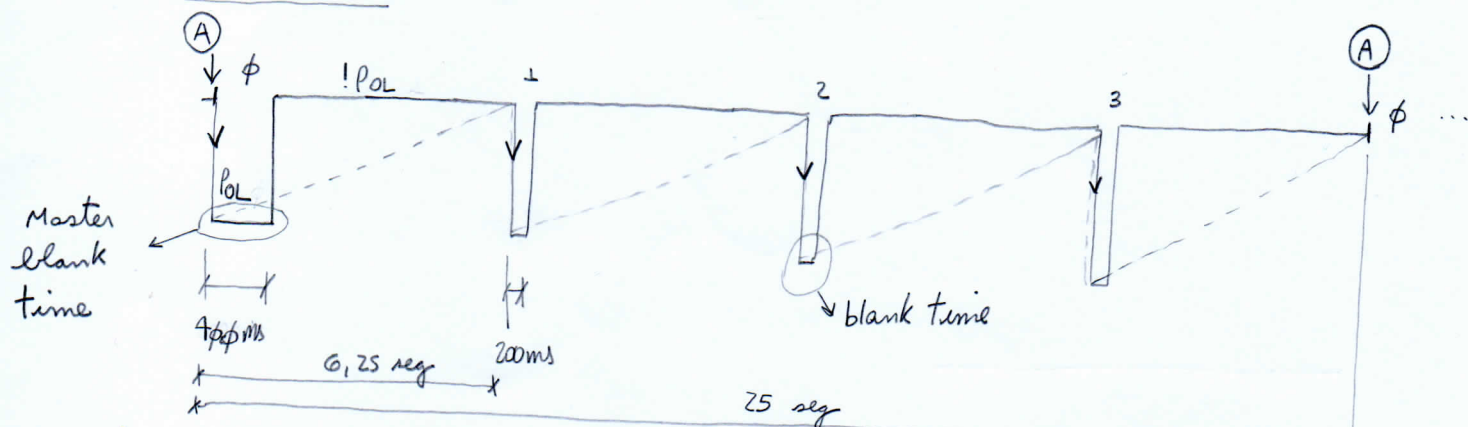
- Desenvolver um Sync IP, inserido no Platform Designer. (Desabilitar os módulos PIO Sinc-in e Sinc-out que foram usados p/ testes)
- clock : 50 MHz

C). Diagrama de blocos :



D). Requisitos / features

# • Cartas de Tempo - sinal de sincronismo (p/ Generator)



• O sinal de sinc. pode ser tratado como um PWM.  $\rightarrow$  P/O

Há um registro de configuração de período (counter principal), um registro de "duty cycle" (master blank time @ 400 ms default NFEE), um registro de blank time (normal @ 200 ms default NFEE), e um registro que diz o número de ciclos de um 'macro' ciclo (Na carta acima,  $N=4 \Rightarrow \phi \dots 3$  ciclos).

• O Master blank time só é mais largo no N-FEE mode. No F-FEE a sua largura é igual aos demais blank times.

N-FEE : Master blank time : 400 ms

blank times : 200 ms

Ciclo : 6,25 seg  $\rightarrow$  6,25 .. 15 seg

Macro ciclo : 25 seg  $\rightarrow$  25 .. 60 seg

$N=4$

Valores default. São configuráveis.

F-FEE : Master blank time : 200 ms

blank times : 200 ms

Ciclo : 2,3 seg

Macro ciclo : 2,3 seg  $\rightarrow$  mín.: 1,0 seg ; máx. = 60 seg

$N=1$

• O sync IP possuirá registros "flexíveis" p/ a construção da carta de tempos. É papel da CPU Nios calcular os parâmetros necessários para os modos N-FEE/F-FEE e configurar o sync IP.

• Deve haver [um registro]  $\rightarrow$  control p/ reset do módulo. Em reset, o IP deve manter o sync-gen. em '1'.

ou em !POL

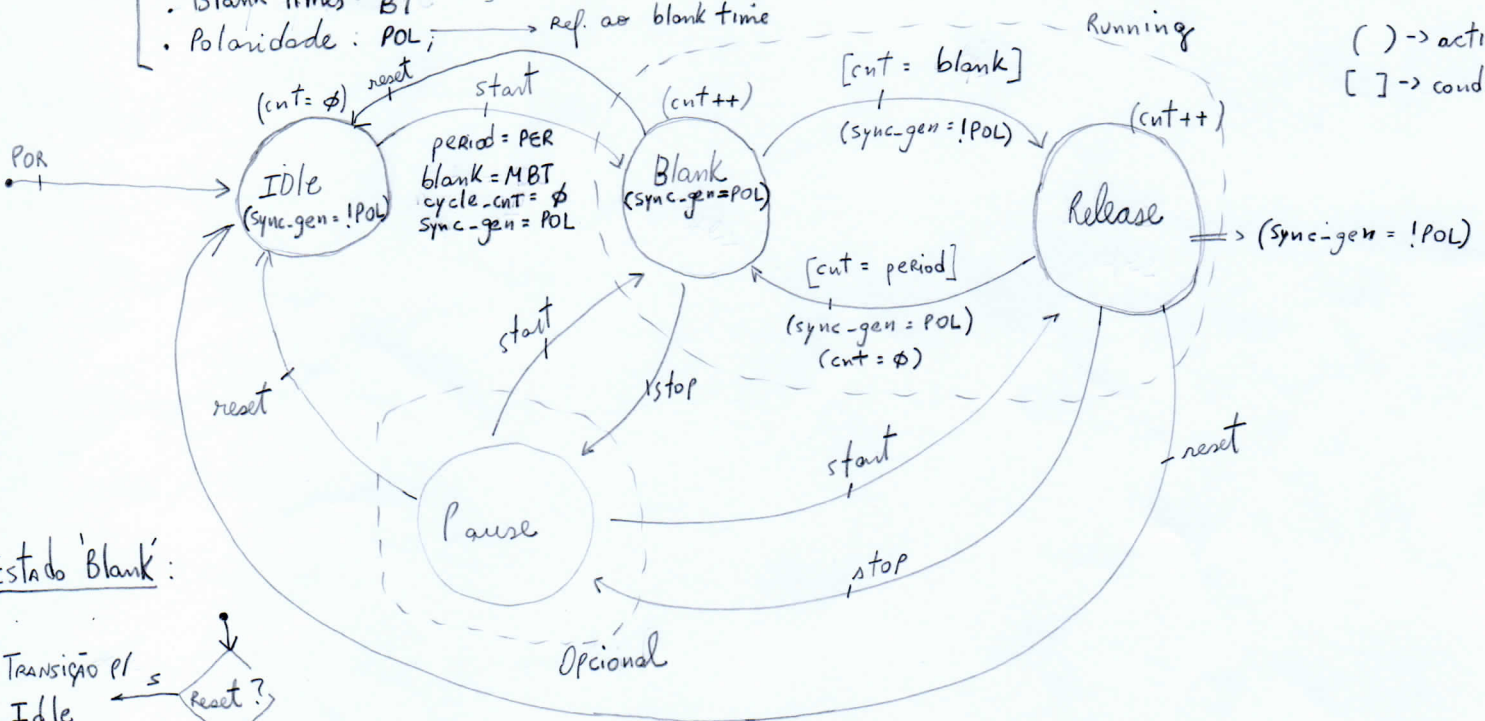


• Deve haver, ainda, algum registro <sup>e controle</sup> p/ promover 'erros' intencionais na geração do sync. Exemplos:

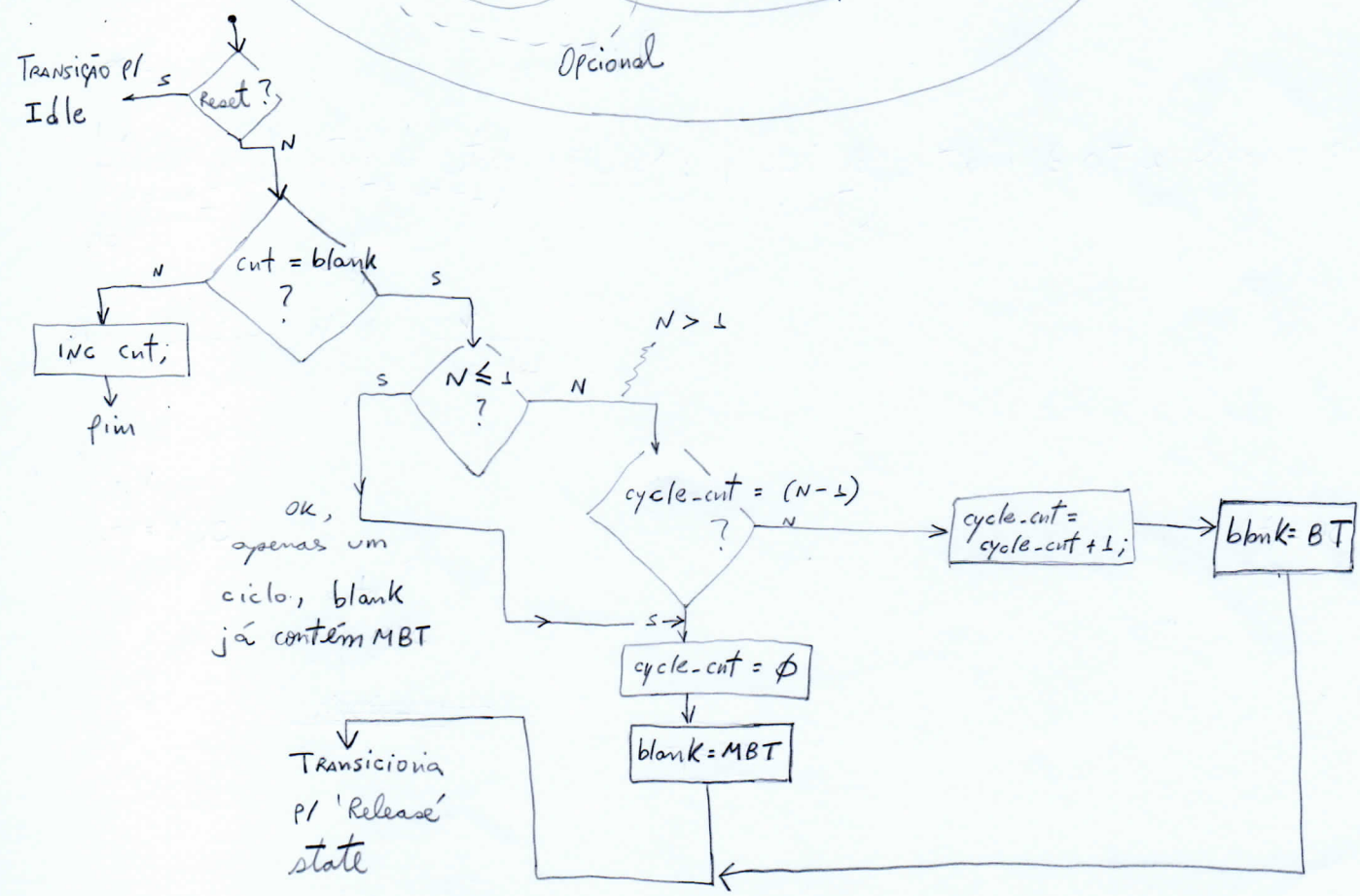
- 1 bit de 'missing sync': Se '1' → o gerador suprime um pulso de sync.
- 1 bit de 'wider sync': Se '1' → o gerador 'alarga' um dos pulsos de sync.
- ...

### E) Diagrama de estados - Unid. de controle - fora no gerador interno

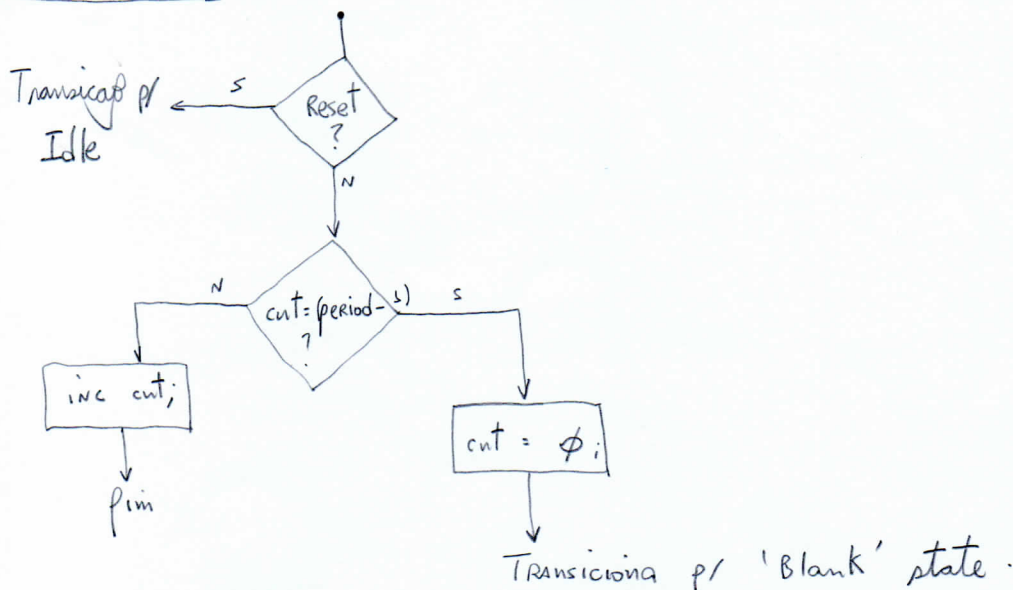
- Número de ciclos:  $N$ ;
  - Contador principal: cnt ( $\overset{32}{n}$  bits);
  - Período: PER;
  - Blank times: MBT;
  - Polaridade: POL;
- parâmetros; vars/signals: period; blank; cycle-cnt.
- ( ) → action  
[ ] → condition



### Estado 'Blank':







## F) Registros :

• Config :

- ERROR-injection  $D \rightarrow A$
- OST  $\rightarrow$  One shot time  $D \rightarrow A$
- MBT  $\rightarrow$  Master blank time
- BT  $\rightarrow$  Blank time
- PER  $\rightarrow$  período
- N  $\rightarrow$  n. de ciclos do macro ciclo (1..K)
- POL  $\rightarrow$  Polaridade do sync-gen

x base de 1/50MHz

• Control :

- Start  $D \rightarrow A$
- Reset  $D \rightarrow A$
- Int-EXT  $D \rightarrow C$
- One-shot  $D \rightarrow A$
- Err-inj  $D \rightarrow B$
- EN-CH-A
- EN-CH-B
- EN-CH-C
- EN-CH-D
- EN-CH-E
- EN-CH-F
- EN-CH-G
- EN-CH-H
- EN-SYNC-OUT

$\Rightarrow$  '0'  $\rightarrow$  sync off na saída respectiva  
'1'  $\rightarrow$  sync ON "

• Interrupt out (to CPU master) :

idle : 0 one-shot: 2  
Running: 1 Err-inj: 3

$A \rightarrow D$  : State : 8 bits  
 $D$  : Int-ext-n : 1 bit  
 $A \rightarrow D$  : Cycle-number : 8 bits  
 $D$  : ERROR : 8 bits (code)  $\rightarrow$  TBD.

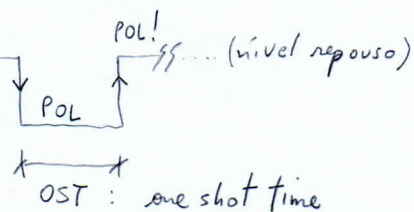
• ERROR  $D$

- Master blank pulse  $\rightarrow$  Cada uma: Enable Flag 2 bits
- Blank pulse  $\rightarrow$  1 só: Blank pulse (serve p/ os estados running, one-shot, error-inj.)
- Reset  $D \rightarrow X$

## G) Complementos na lógica original

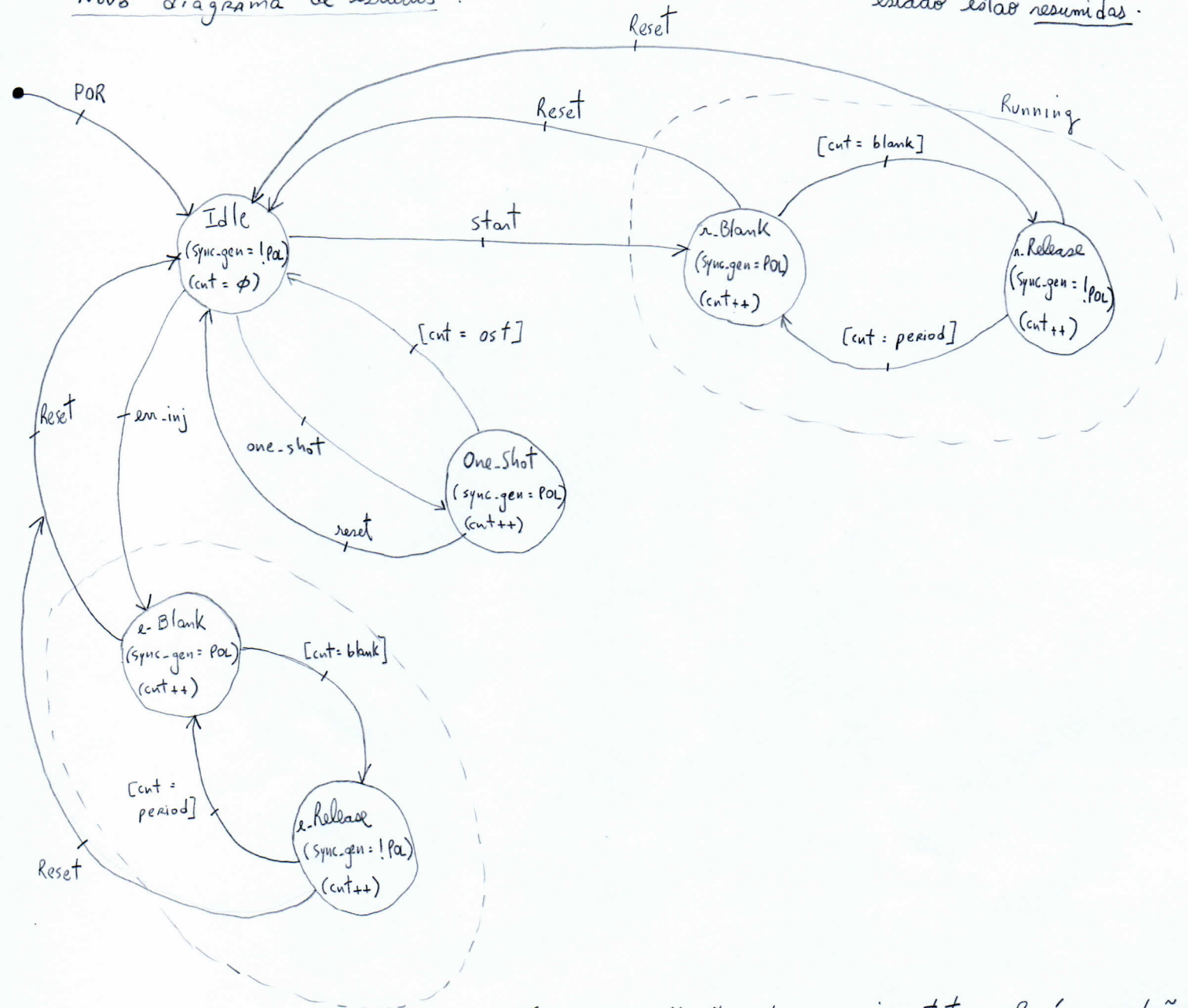
• Serão inseridos dois estados na máquina geradora: one-shot e error-injection.

\* One-shot : gera um blank pulse único:









\* ERROR-injection : Pode ser encarado como uma replicação do macro estado "running". Só que, dependendo do registro de error-injection, 'falhas' são emuladas na carta de tempos.

- Novo diagrama de estados:



ERROR.inj  $\rightarrow$  É uma replicação do running state. Porém, poderão ser inseridos erros no sinal sync-gen, conforme o registro sync-error-injection.

14/11/18 - TODO's :

- out!  Revisão final dos fontes VHDL ;
- out!  Atualização <sup>dos</sup> registros de status ;
- out!  Atualização das flags de interrupção ;
- ad!  Testbench ;
- ad!  Simulação em Modelsim ;
- out!  CRIAR component INTERRUPT ;

Gerador: ✓  
status-0.state ✓  
status-0.cycle.number ✓

rise\_0 . blank\_pulse\_rise . ☒   
 ✓(check enable)

Unid. controle : ✓

- status-register, int-ext n
- status-register, error-code
- interrupt-register.
- error-ist-p/og
- (decar enable)

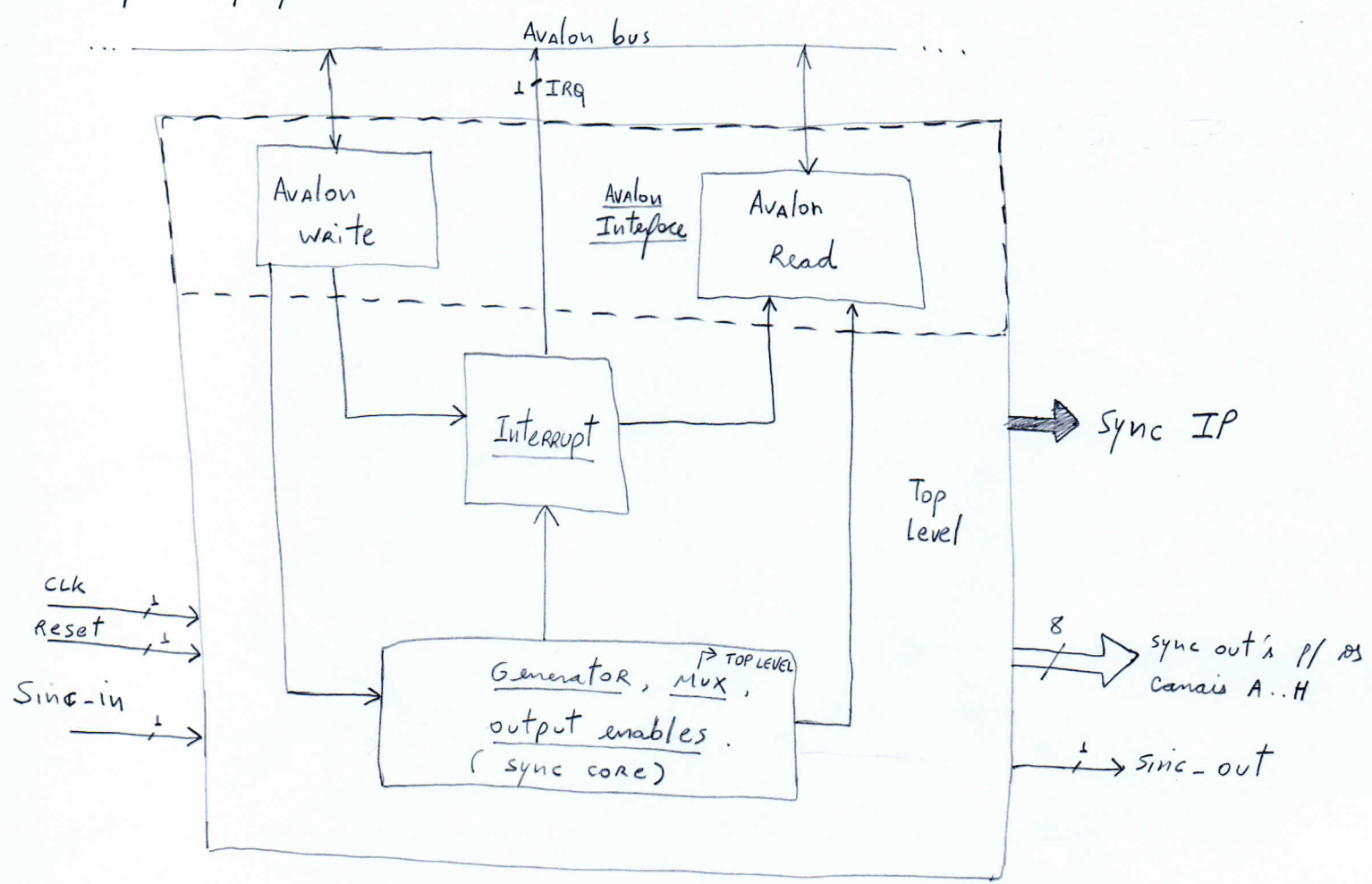
Pendente:  
ISR Flags clear!

Resolvido c/  
módulo "int"

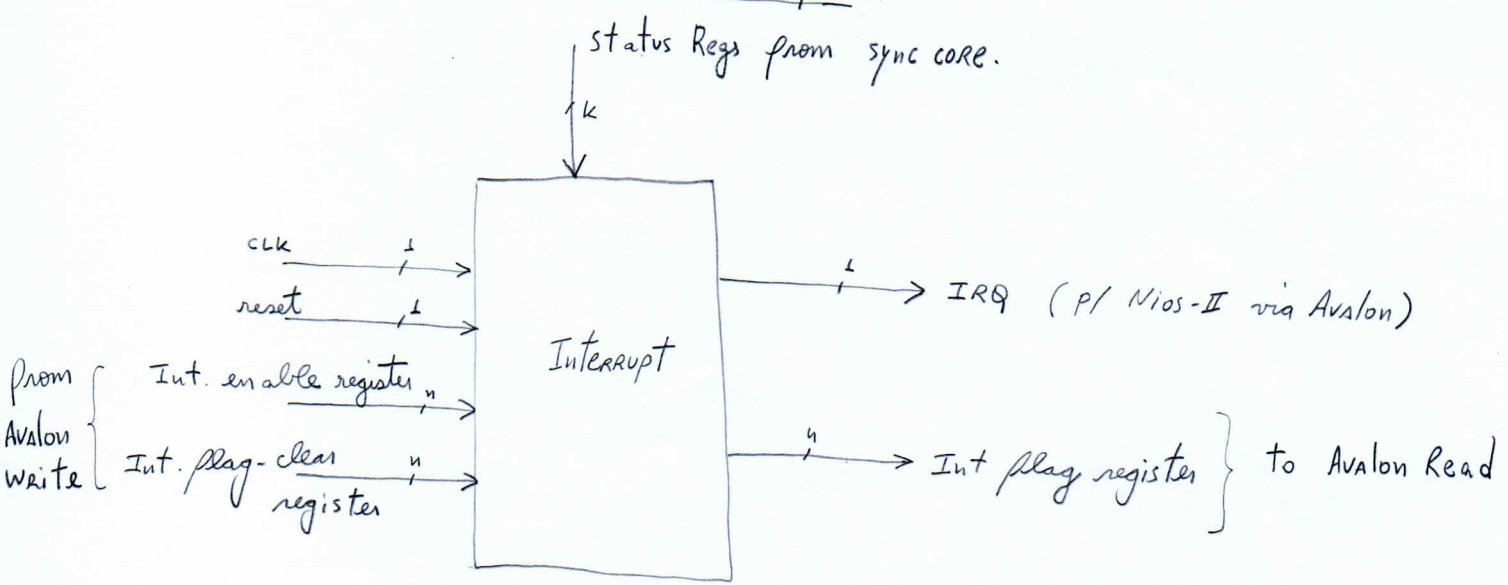
TBD

- Criação do IP em Platform Designer;
- Criação do driver em C P1 Nios-II; (API)
- Validação real no Simulador completo.

- Observação importante : visando um melhor reaproveitamento de código e maior modularização, a parte de interrupções será desmembrada em um componente próprio :



- Diagrama de blocos - componente interrupt :



- Toda a parte de interrupção deve ser removida do sync core e passada ao component interrupt.

O sync core só se preocupa em produzir os seus registros de status. Estes serão lidos pelo interrupt component, para então gerar os sinais de interrupção.



## • Lógica do módulo 'int' :

7/9

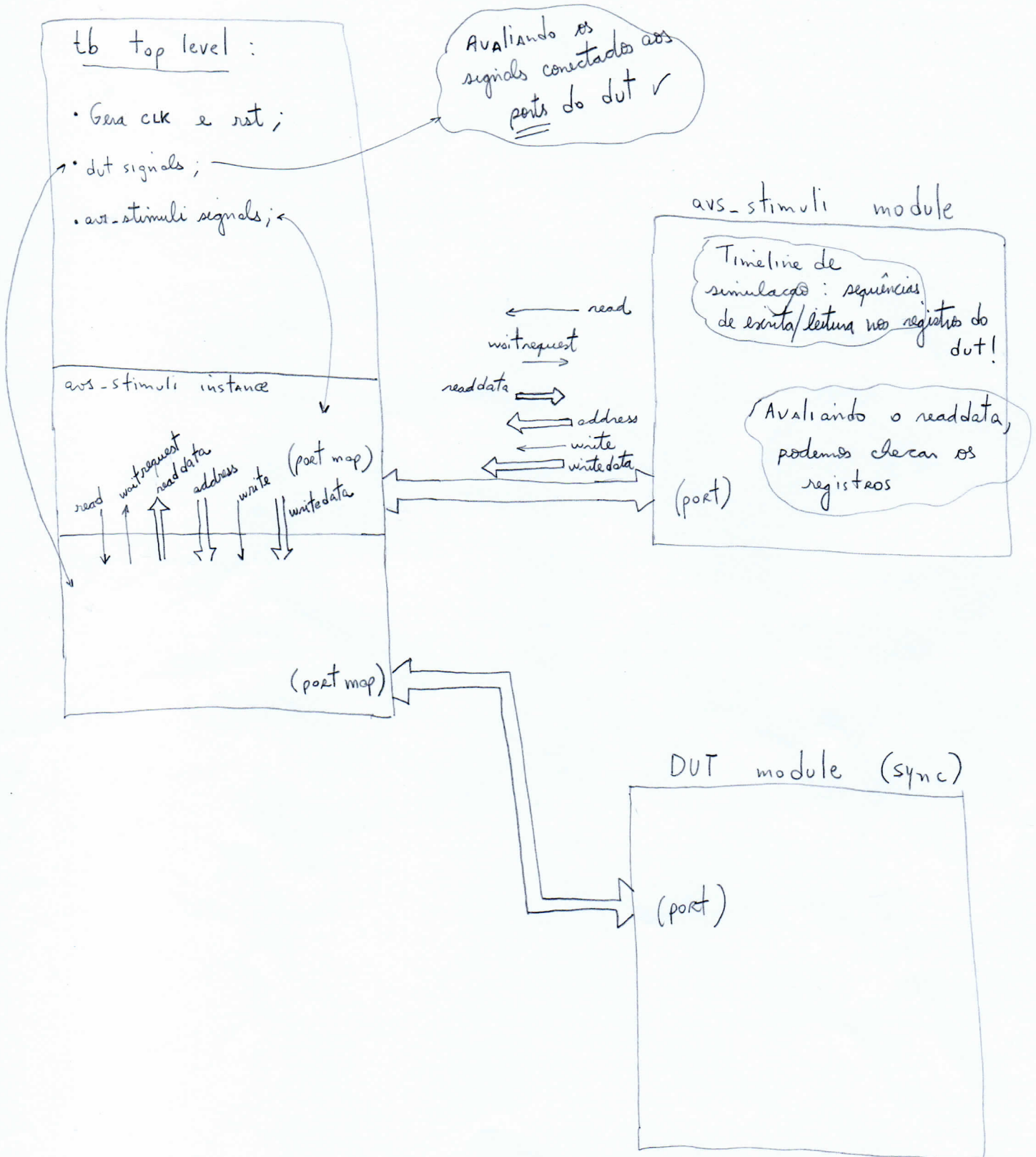
Processo:

- Se reset  $\rightarrow$ 
  - Zerar flags de int;
  - Zerar signals internos;
  - Zerar IRQ.
- Se não, a cada clk :
  - A. Se (flag clear de int erro = '1')  $\rightarrow$  Limpa flag erro int
  - Se não:
    - Se: (erro.code  $\neq \emptyset$ )  $\rightarrow$  Ativa flag int se enable ok ✓
    - else:
      - (Se erro.code =  $\emptyset$ )  $\rightarrow$  fim // desativa flag e
  - B. Repete bloco analisando a flag de blank pulse ✓  
(testa se o sinal sync pós mud está em blank level)
  - C. • IRQ  $\Rightarrow$  Se alguma flag de int for '1'  $\Rightarrow$  default irq pol. ( 1 )  
Se não  $\Rightarrow$  not default irq polarity. ( 0 ).

$\hookrightarrow$  Se sim, e enable ok  $\rightarrow$  1  
zera flag. Se não  $\rightarrow$  solve flag. <sub>INT</sub>

# Arquitetura do testbench

8/9





# Sync Simulation

clock: 20ms  $\rightarrow$  50MHz  $\checkmark$

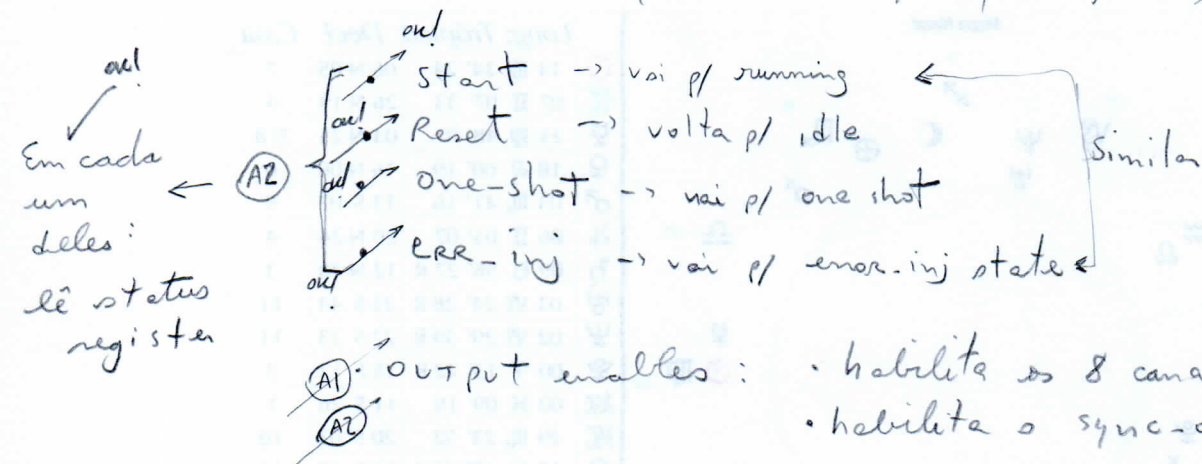
9/9

Modelsim - Simulation time = 150 $\mu$ s  $\checkmark$

- al
- A) • Sem error injection;  
• Sem Interrupts;

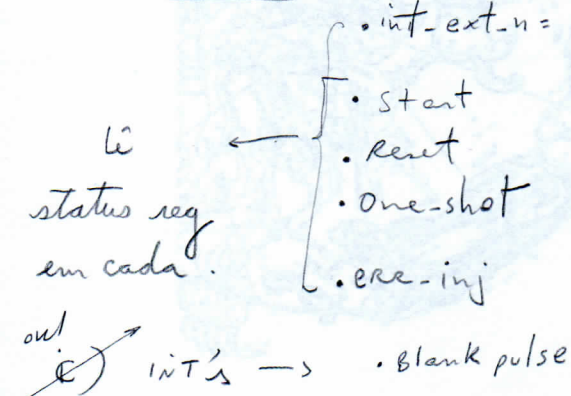
• Config : N-FEE sync: MBT: (400ns) = 20  $\checkmark$   
BT: (200ns) = 10  $\checkmark$   
PER: (6,25 $\mu$ s)  $\approx$  312  $\checkmark$   
OST: (100ns) = 5  $\checkmark$   
general. number-of-cycles = 4  $\checkmark$

• Control : • int-ext-n = 0  $\Rightarrow$  ext. sync first / int. sync after.



B) • Config : F-FEE sync: MBT: (200ns)  $\rightarrow$  Usa 400ns p/ ter certeza!  
BT: (200ns)  $\rightarrow$  P/N = 1  $\Rightarrow$  vale o MBT!  
PER: (2,3 $\mu$ s)  
OST: (500ns)  
general. number-of-cycles = 1  $\leftarrow$  out out

• Control : • int-ext-n = 1 (int. sync)



D) Tentativas de reconfig. fora do idle state.