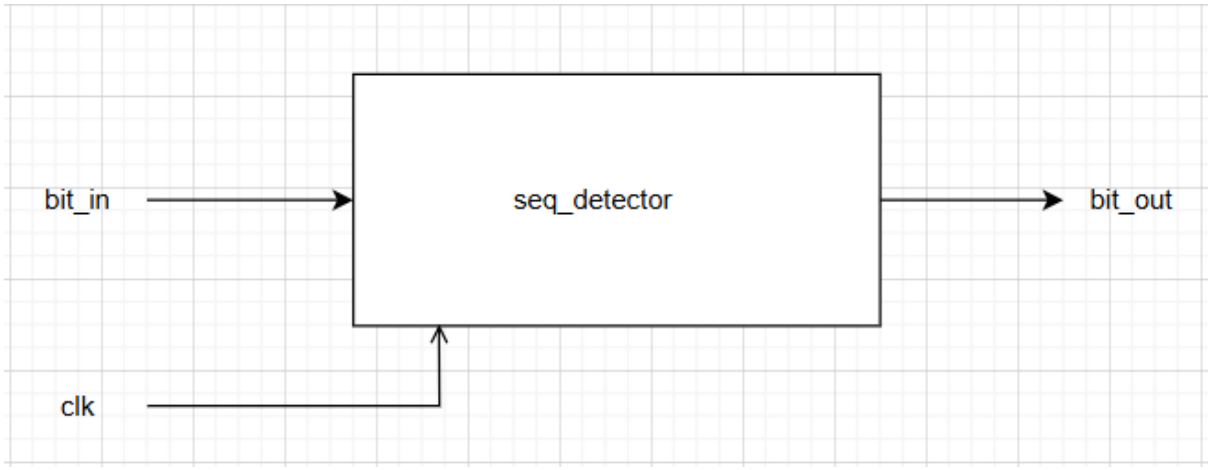
Trabalho 1

Detenção de sequência

Este trabalho envolve o desenvolvimento de um detetor de sequência de bits, ou seja, existirá um recetor de sinal de bit, que estará em ritmo de relógio e para cada levantamento do clock é analisado o bit de entrada, caso a respetiva transição de bits forme a sequência desejada, o bit de saída irá transmitir valor de bit a 1.

Para este trabalho usaremos a respetiva entidade, apresentada pela imagem abaixo e o respetivo código vhdl:



Representação da entidade sq_dectector

```
entity seq_detector is
    Port (
        clk : in std_logic;
        rst : in std_logic;
        x   : in std_logic;
        z   : out std_logic
    );
end seq_detector;
```

Entidade seq_detector em vhdl

A arquitetura para este projeto será do tipo *Behavioral* visto que será utilizado processos de forma a correr sequencialmente.

O processo principal terá a lógica de interpretar o sinal de entrada e verificar, consoante o estado atual se o mesmo corresponde a valor correto da sequência ou não.

Com isto é necessário determinar quando estados são necessários até alcançar a sequência desejada. Para este trabalho será a usado a sequência de bits '100101' com um total de 6 bits, logo é necessário declarar 6 estados, que representam até onde a sequência de bits de entrada é igual a sequência desejada.

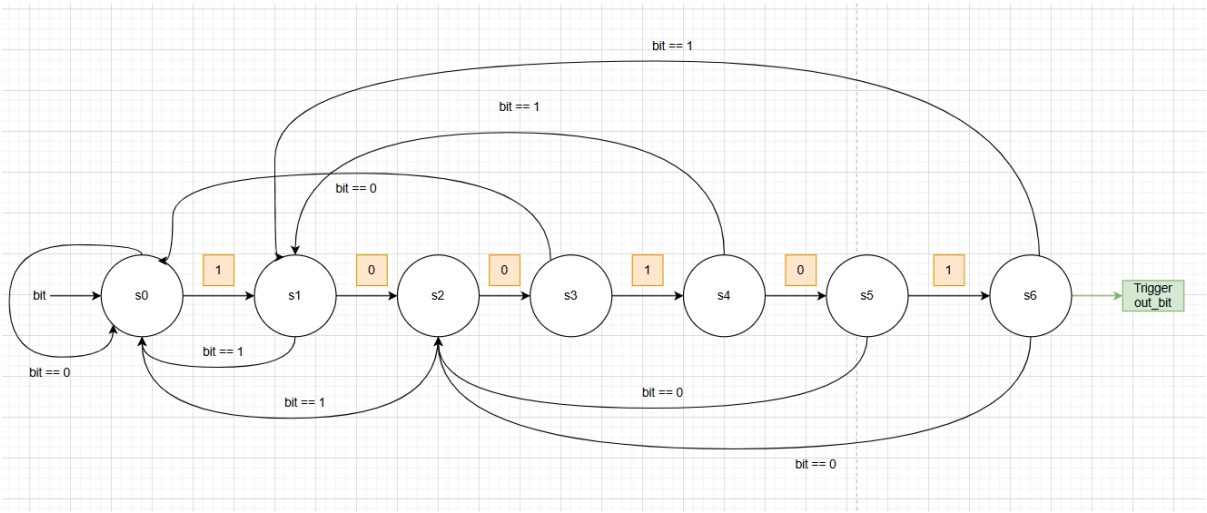
Por exemplo, se sequencia de entrada for "100", então o estado atual da nossa entidade é o estado 3.

Esta representatividade em *vhdl pode* ser feita usando *type*, que na prática é representação do estado em palavras, apenas visível no código, que após compilado o mesmo é convertido para bits que representam um estado, por exemplo, o estado 1 seria igual a '00', o estado 2 seria igual '01', por assim adiante.

```
type state_type is (s0, s1, s2, s3, s4, s5, s6);
```

Declaração de estado em vhdl

Alteração de estado prosseguirá uma determinada lógica que está definida na imagem abaixo.



Lógica para deteção da sequência '100101'

Consoante a imagem acima percebemos que existe uma verificação a cada bit de entrada, caso coincide com parte da sequência avançará apara o próximo estado caso contrário voltará para o estado a qual ainda corresponde a sequência ponderando os bits anteriores.

A volta para estados diferentes em caso de não existir sequência, tem objetivo de aproveitar uma subsequência que estava integrada na sequência original.

Após chegada ao estado 6 é necessário colocar o out_bit '1', para isso é criado um signal que referência o estado atual e caso o estado não seja o 6 então será dado o valor de '0' ao bit de saída caso contrário será dado valor '1' ao bit de saída.

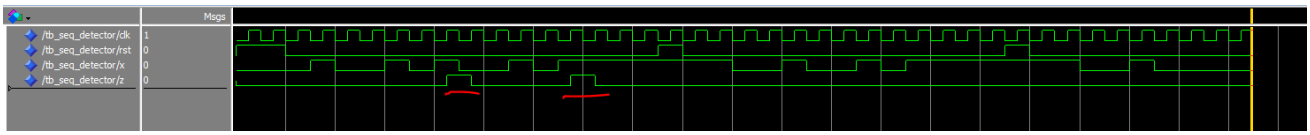
```
-- Moore output: depends only on the state
z <= '1' when state = s6 else '0';
```

Atribuição de valor do out_bit

Nota-se que out_bit apenas ficará a '1' um ciclo de relógio pois o estado é sempre atualizado a cada ciclo de relógio, sendo impossível seguindo a lógica mencionada, estar presente no estado 6 continuamente.

Simulação

A imagem abaixo demonstra a simulação do nosso projeto detetor de sequência.



Simulação no ModelSim

É observável que na linha "z", o bit é elevado a 1 duas vezes, o que nos indica que foi observado duas vezes a sequências, e se traduzimos as entradas de bit na linha x temos a seguinte combinação.

10010100101

Separando em cores vemos que a sequência for reproduzida juntamente com os bits a cor amarela e o bit com cor azul, e seguidamente uma outra sequência reproduzida com cor azul e verde, reutilizando o mesmo bit na deteção anterior, neste caso o bit a azul.

Seguindo o resto do processo da simulação observamos que mais nenhuma sequência foi detetada.