ISEL

DEETC

Departamento de Engenharia Electrónica e de Telecomunicações e de Computadores

Laboratório de Informática e de Computadores

2021 / 2022 verão

LEIC

Etapas de projeto de uma Máquina de Estados

- Descrição do problema
- Especificação do sistema
- Arquitetura do sistema
- Implementação
  - Lógica Programável

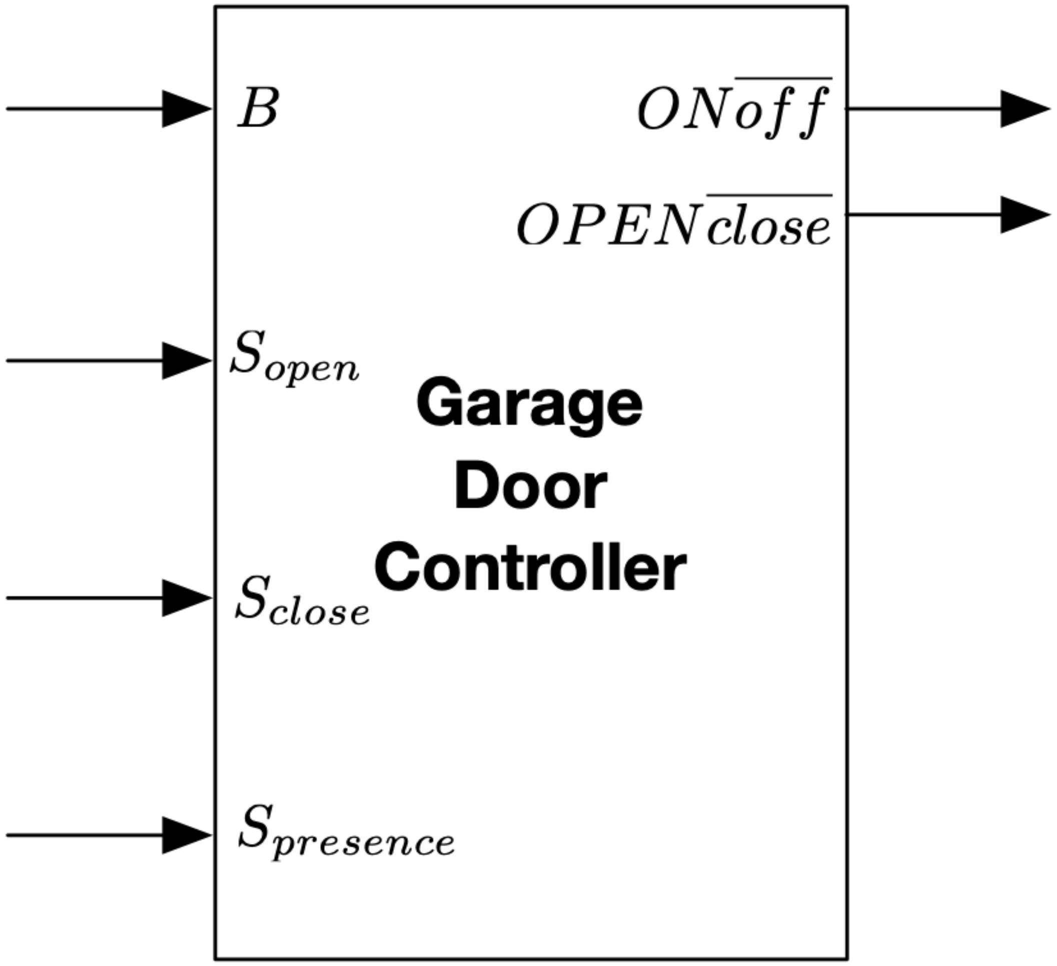
Projeto: Controlo de Portão

- Descrição do problema
  - Projecte e realize um sistema de abertura e fecho de um portão
  - Considera-se que o sistema tem um motor e um botão para abertura e fecho.

Projeto: Controlo de Portão

(*Garage Door Controller*)

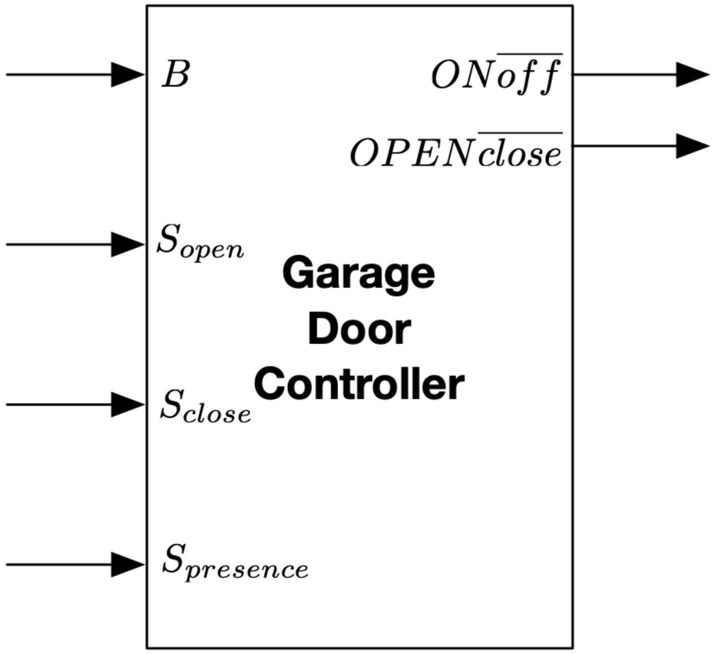
- Especificação
  - Botão ***B***
  - Dois Sensores fim de curso, $\boldsymbol{S_{open}}$ e $\boldsymbol{S_{close}}$
  - Um sensor de obstrução, $\boldsymbol{S_{presence}}$
  - Motor com dois sinais de controlo, ON/off, OPEN/close

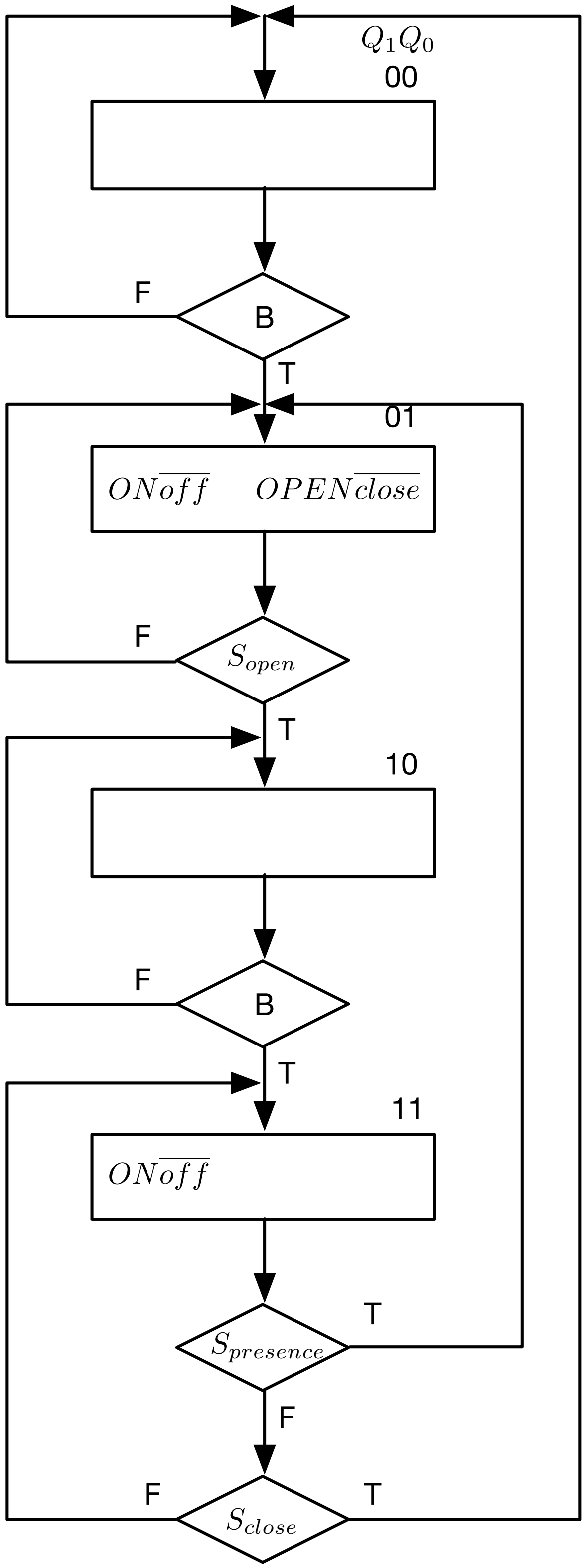


Projeto: Controlo de Portão

(*Garage Door Controller*)

- Arquitetura do sistema





Close

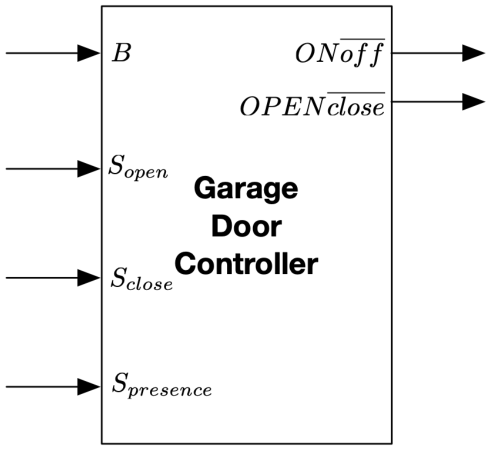
Opening

Open

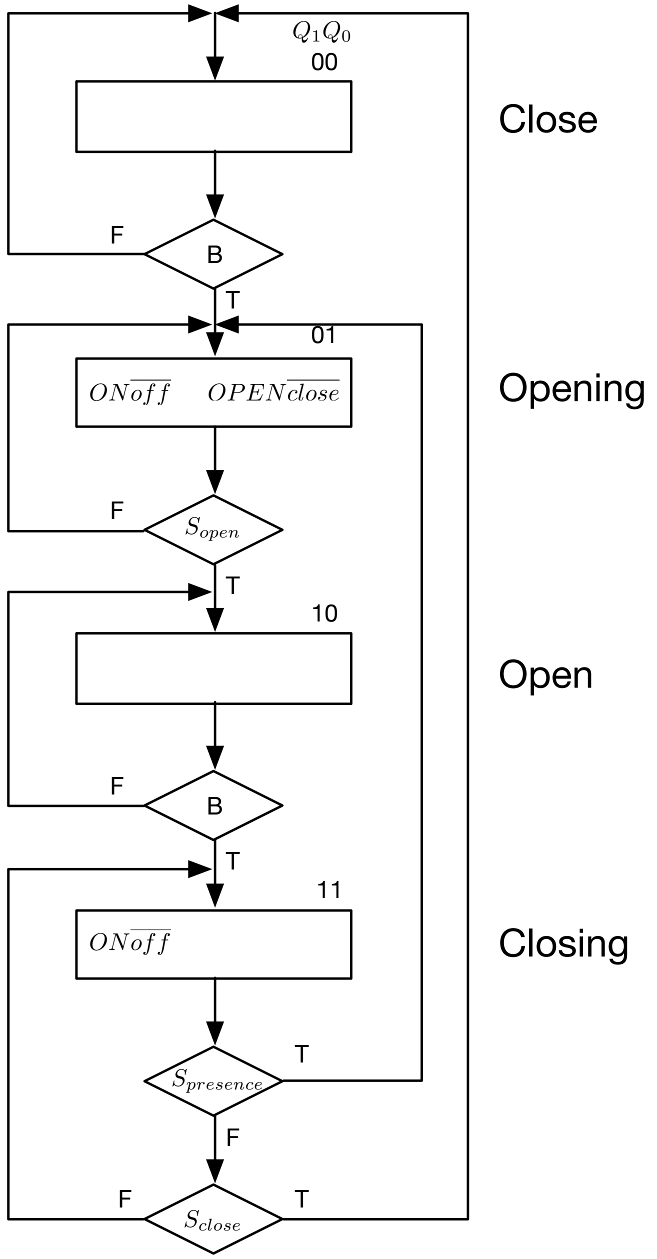
Closing

Projeto: Controlo de Portão

(*Garage Door Controller*)



- Projeto ASM-Chart

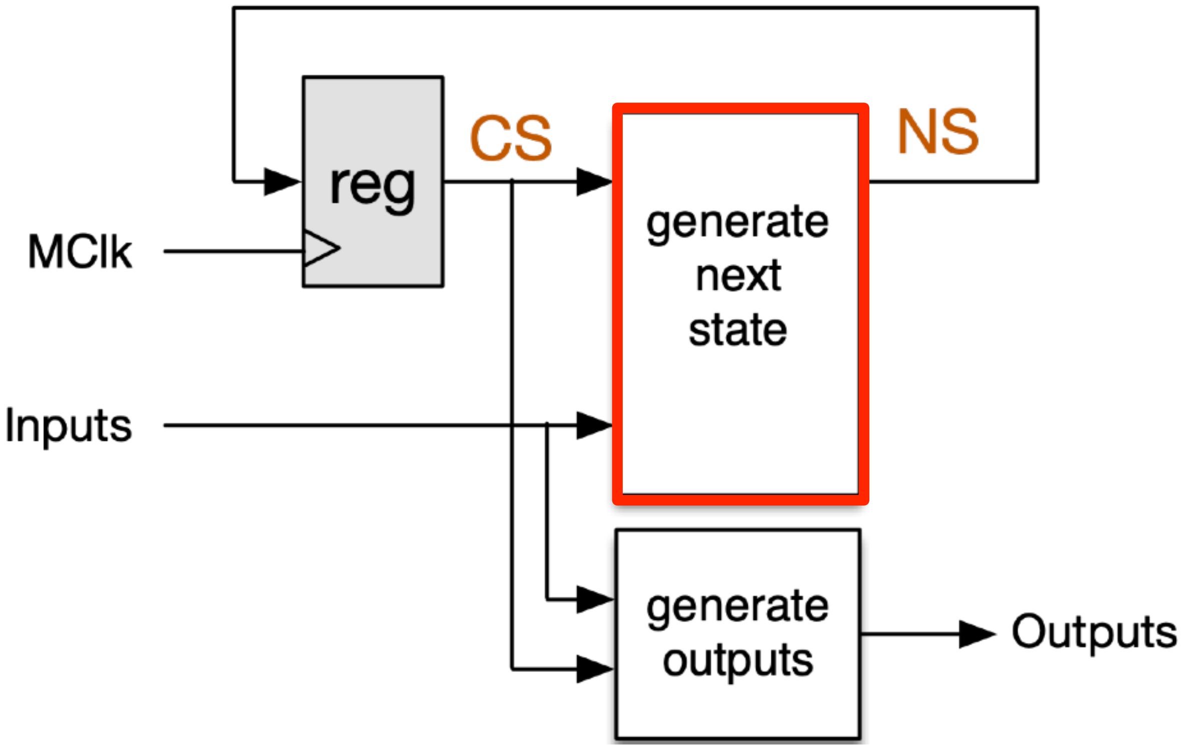


***State* =**

		Q_0
	Close	Opening
Q_1	Open	Closing

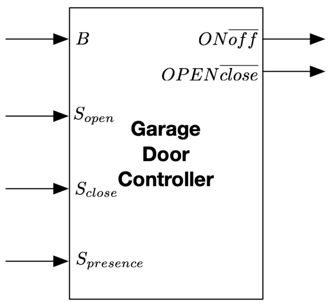
CS = Current State

NS = Next State

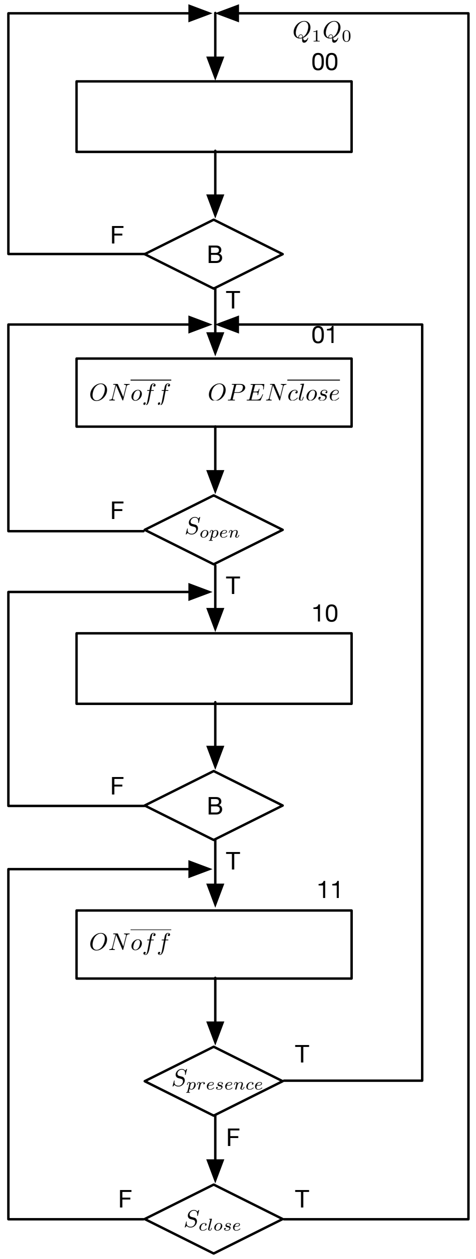


Projeto: Controlo de Portão

(*Garage Door Controller*)



- Projeto ASM-Chart



Close

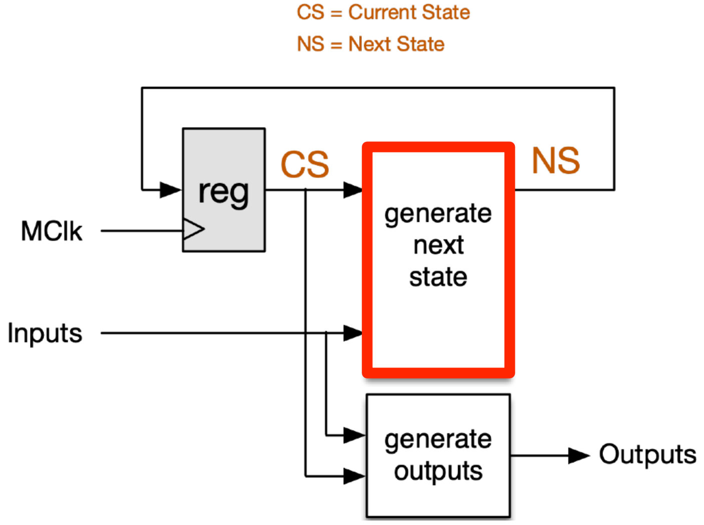
Opening

Open

Closing

State =

	Q_0	
Close	Opening	
Open	Closing	Q_1



$D_1 =$

	Q_0	
0	S_{open}	
1	(F_1)	Q_1

$(F_1)\ \overline{S_{presence}} \cdot \overline{S_{close}}$

$D_0 =$

	Q_0	
B	$\overline{S_{open}}$	
B	(F_2)	Q_1

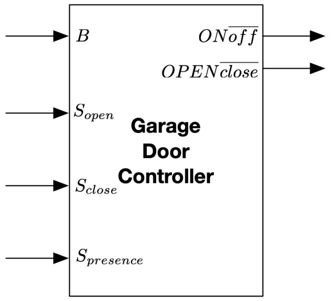
$(F_2)\ S_{presence} + \overline{S_{close}}$

$$\begin{aligned} D_1 &= \overline{Q_1} \cdot Q_0 \cdot S_{open} + \\ &+ Q_1 \cdot \overline{Q_0} + \\ &+ Q_1 \cdot Q_0 \cdot \overline{S_{presence}} \cdot \overline{S_{close}} \end{aligned}$$

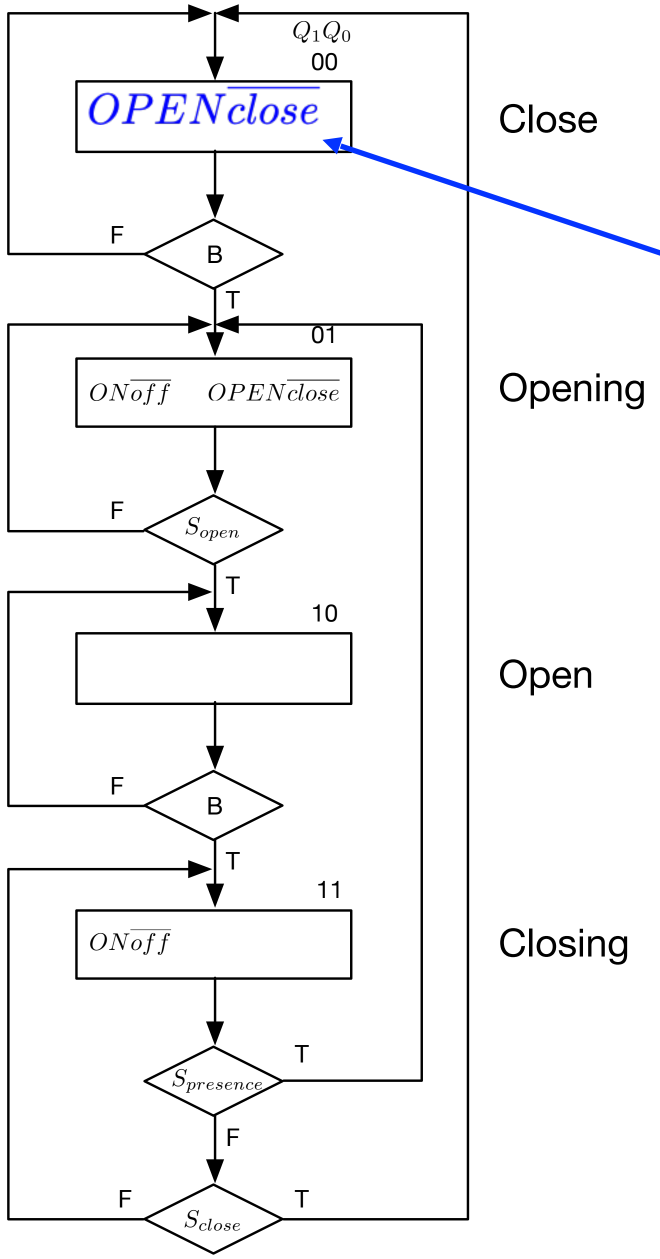
$$\begin{aligned} D_0 &= \overline{Q_0} \cdot B + \\ &+ \overline{Q_1} \cdot Q_0 \cdot \overline{S_{open}} + \\ &+ Q_1 \cdot Q_0 \cdot (S_{presence} + \overline{S_{close}}) \end{aligned}$$

Projeto: Controlo de Portão

(*Garage Door Controller*)

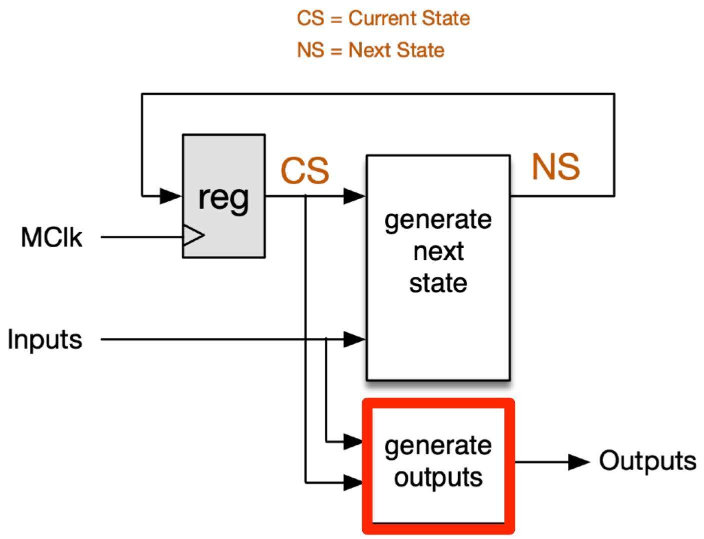


- Projeto ASM-Chart



State =

	Q_0	
Close	Opening	
Open	Closing	Q_1



$ON\overline{off}$ =

	Q_0	
0	1	
0	1	Q_1

$OPEN\overline{close}$ =

	Q_0	
1	1	
—	0	Q_1

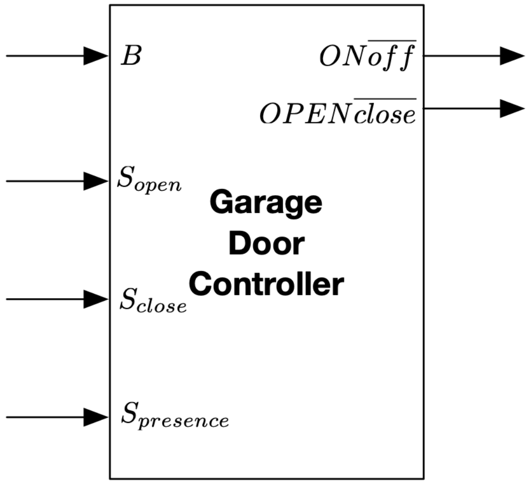
$$ON\overline{off} = Q_0$$

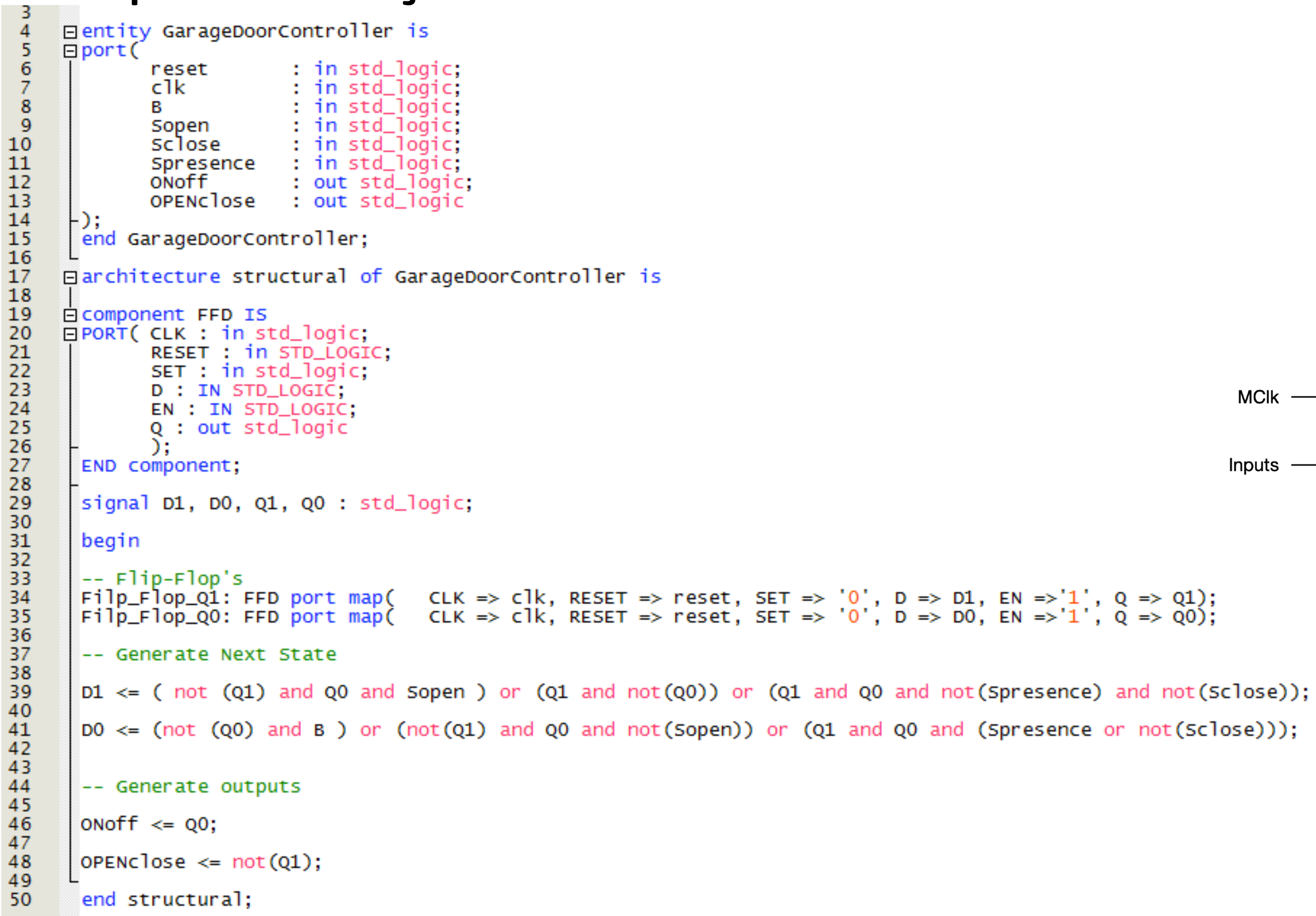
$$OPEN\overline{close} = \overline{Q_1}$$

Projeto: Controlo de Portão

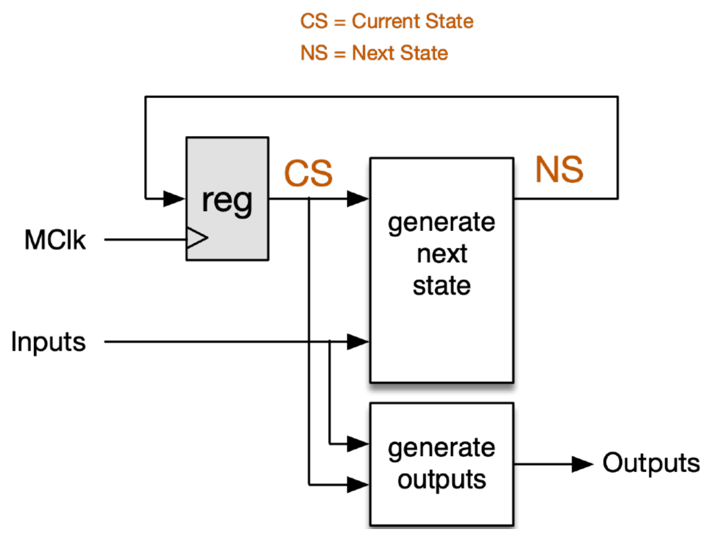
(*Garage Door Controller*)

- Implementação ASM-Chart





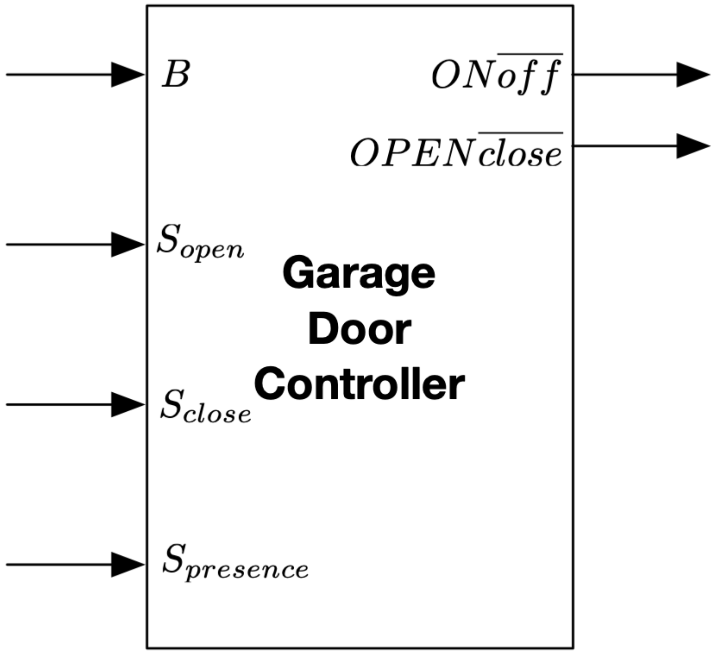
```vhdl

entity GarageDoorController is
port(
      reset            : in std_logic;
      clk              : in std_logic;
      B                : in std_logic;
      Sopen            : in std_logic;
      Sclose           : in std_logic;
      Spresence        : in std_logic;
      ONoff            : out std_logic;
      OPENClose        : out std_logic
);
end GarageDoorController;

architecture structural of GarageDoorController is

component FFD IS
PORT( CLK : in std_logic;
      RESET : in STD_LOGIC;
      SET : in std_logic;
      D : IN STD_LOGIC;
      EN : IN STD_LOGIC;
      Q : out std_logic
      );
END component;

signal D1, D0, Q1, Q0 : std_logic;

begin

-- Flip-Flop's
Filp_Flop_Q1: FFD port map(   CLK => clk, RESET => reset, SET => '0', D => D1, EN =>'1', Q => Q1);
Filp_Flop_Q0: FFD port map(   CLK => clk, RESET => reset, SET => '0', D => D0, EN =>'1', Q => Q0);

-- Generate Next State

D1 <= ( not (Q1) and Q0 and Sopen ) or (Q1 and not(Q0)) or (Q1 and Q0 and not(Spresence) and not(Sclose));

D0 <= (not (Q0) and B ) or (not(Q1) and Q0 and not(Sopen)) or (Q1 and Q0 and (Spresence or not(Sclose)));


-- Generate outputs

ONoff <= Q0;

OPENClose <= not(Q1);

end structural;
```

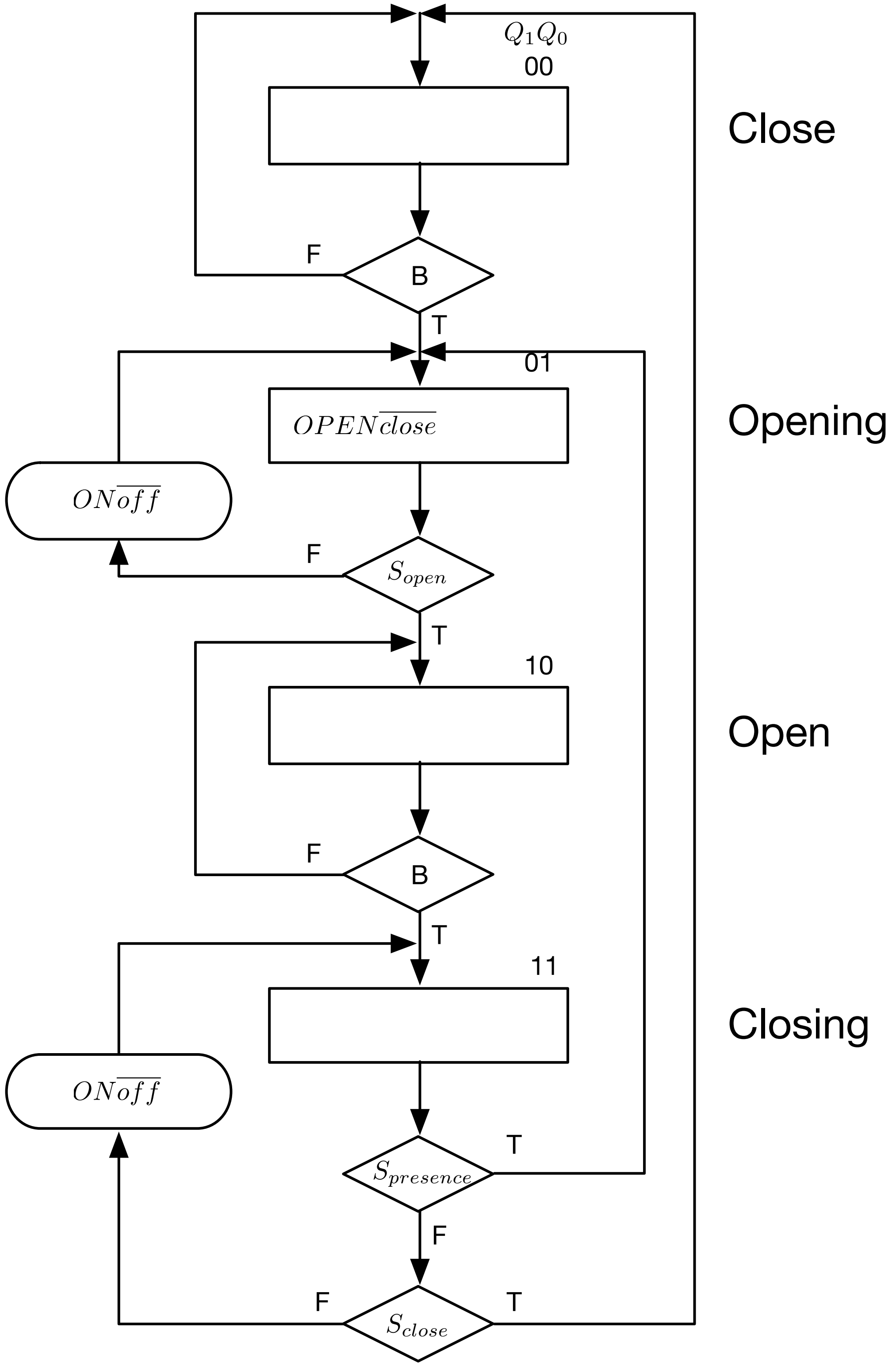


Projeto: Controlo de Portão (2ª versão)

(*Garage Door Controller*)

- Arquitetura do sistema

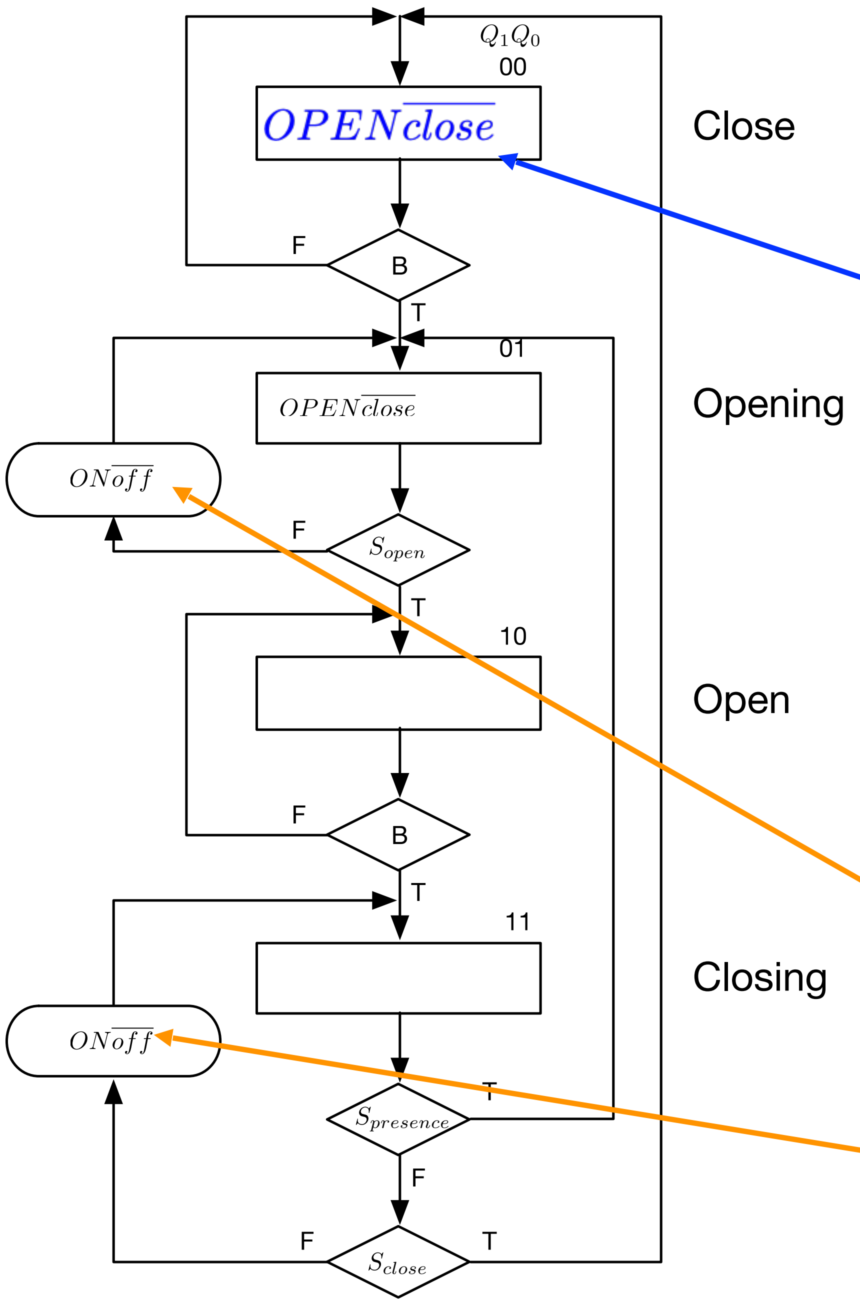




Projeto: Controlo de Portão (2ª versão)

(*Garage Door Controller*)

- Projeto ASM-Chart



State =

$$\begin{array}{c|c|c|c}
 & & Q_0 & \\
\hline
 & \text{Close} & \text{Opening} & \\
\hline
 & \text{Open} & \text{Closing} & Q_1
\end{array}$$

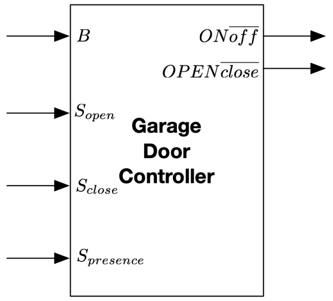
$$ON\overline{off} = \begin{array}{|c|c|c}
 & Q_0 & \\
\hline
0 & \overline{S_{open}} & \\
\hline
0 & (F_1) & Q_1 \\
\hline
\end{array}$$

(F_1) $\overline{S_{presence}} \cdot \overline{S_{close}}$

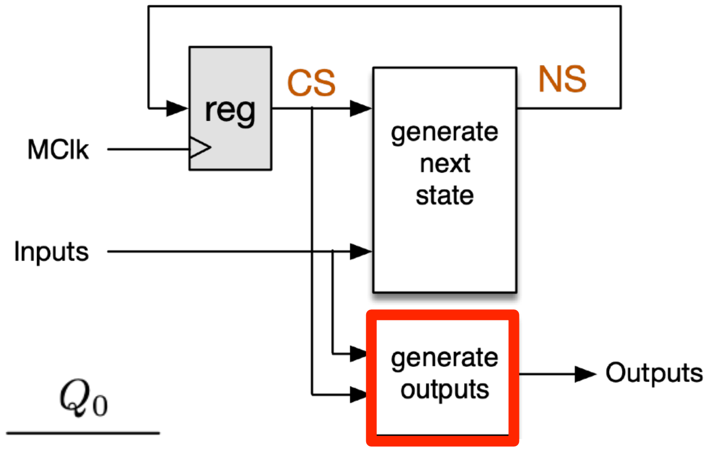
$$OPEN\overline{close} = \begin{array}{|c|c|c}
 & Q_0 & \\
\hline
1 & 1 & \\
\hline
- & 0 & Q_1 \\
\hline
\end{array}$$

$$OPEN\overline{close} = \overline{Q_1}$$

$$ON\overline{off} = \overline{Q_1} \cdot Q_0 \cdot \overline{S_{open}} + Q_1 \cdot Q_0 \cdot \overline{S_{presence}} \cdot \overline{S_{close}}$$



CS = Current State
NS = Next State

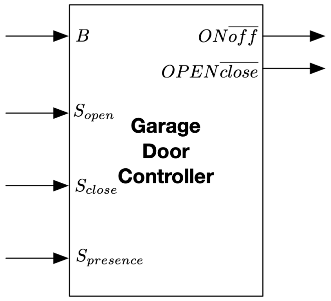


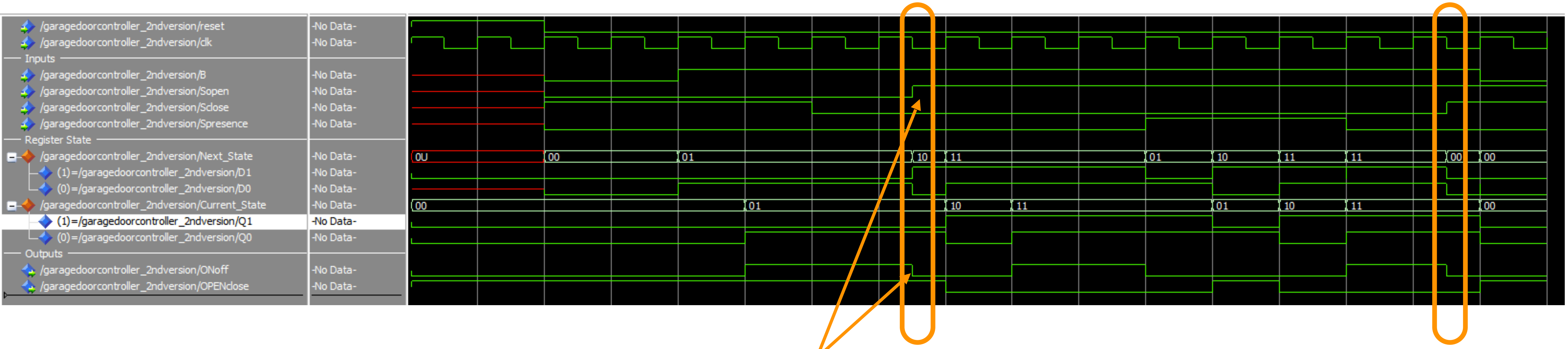
Vantagem desta solução: assim que o sensor de fim de curso fique ativo o motor pára, sem necessitar de mudar de estado

Projeto: Controlo de Portão (2ª versão)

(*Garage Door Controller*)

- Implementação ASM-Chart





/garagedoorcontroller_2ndversion/reset
/garagedoorcontroller_2ndversion/clk
Inputs
/garagedoorcontroller_2ndversion/B
/garagedoorcontroller_2ndversion/Sopen
/garagedoorcontroller_2ndversion/Sclose
/garagedoorcontroller_2ndversion/Spresence
Register State
/garagedoorcontroller_2ndversion/Next_State
(1)=/garagedoorcontroller_2ndversion/D1
(0)=/garagedoorcontroller_2ndversion/D0
/garagedoorcontroller_2ndversion/Current_State
(1)=/garagedoorcontroller_2ndversion/Q1
(0)=/garagedoorcontroller_2ndversion/Q0
Outputs
/garagedoorcontroller_2ndversion/ONoff
/garagedoorcontroller_2ndversion/OPENclose
-No Data-
0U
00
01
10
11
01
10
11
11
00
00
00
01
10
11
01
10
11
00

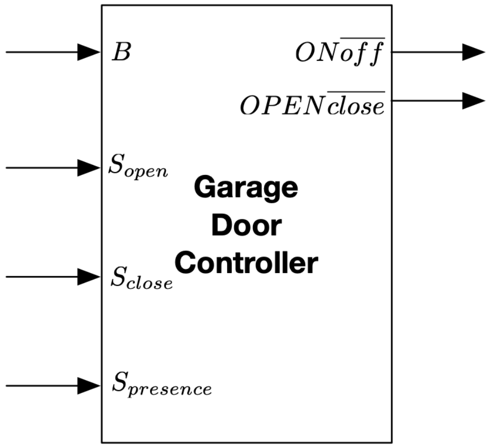
Nesta solução a saída ONoff depende do estado e da entrada, designando-se de saída condicionada

Vantagem desta solução: assim que o sensor de fim de curso fique ativo o motor pára, sem necessitar de mudar de estado

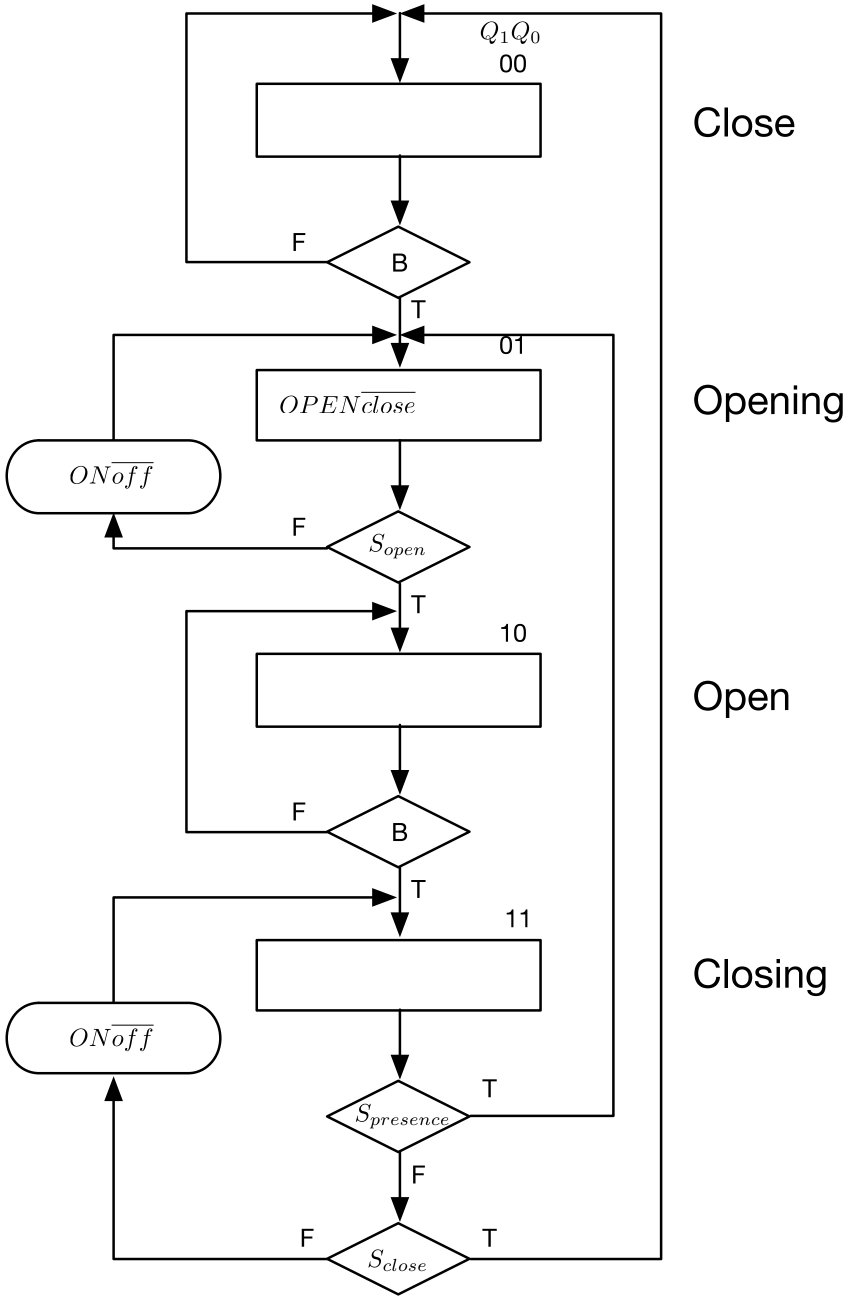
Projeto: Controlo de Portão (3ª versão)

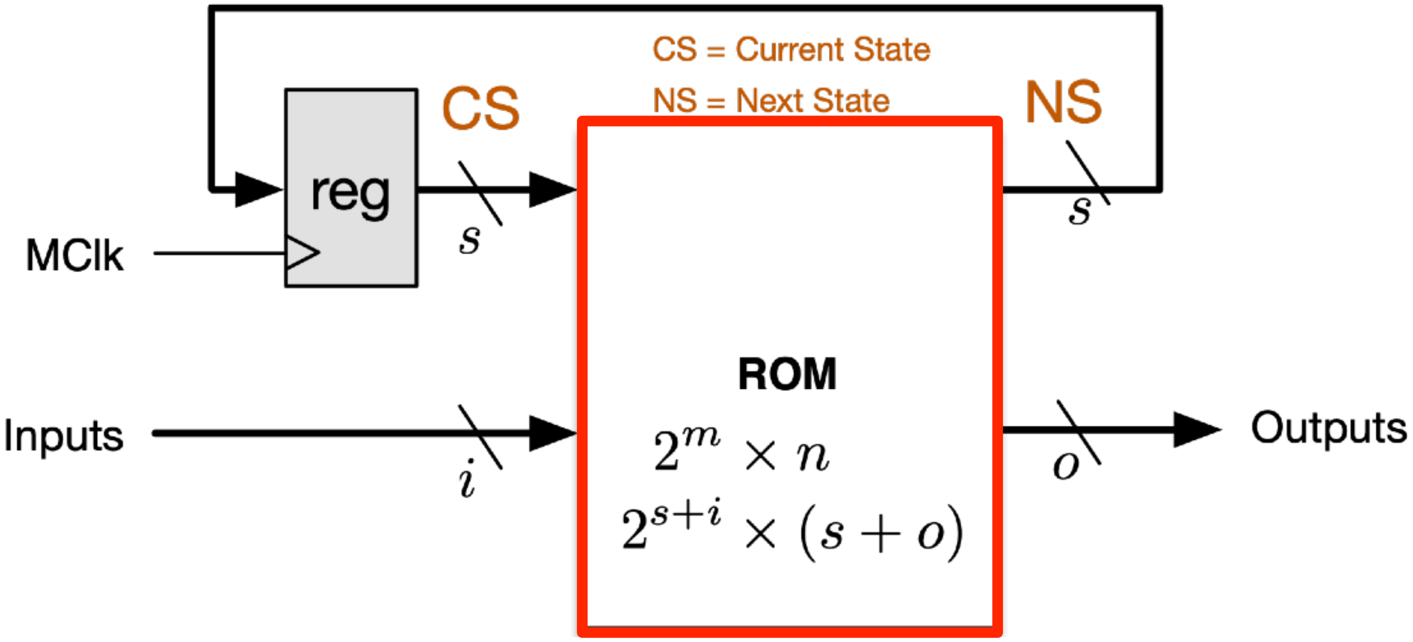
(*Garage Door Controller*)

- Projeto ASM-Chart



		Q_0	
State =	Close	Opening	
	Open	Closing	Q_1

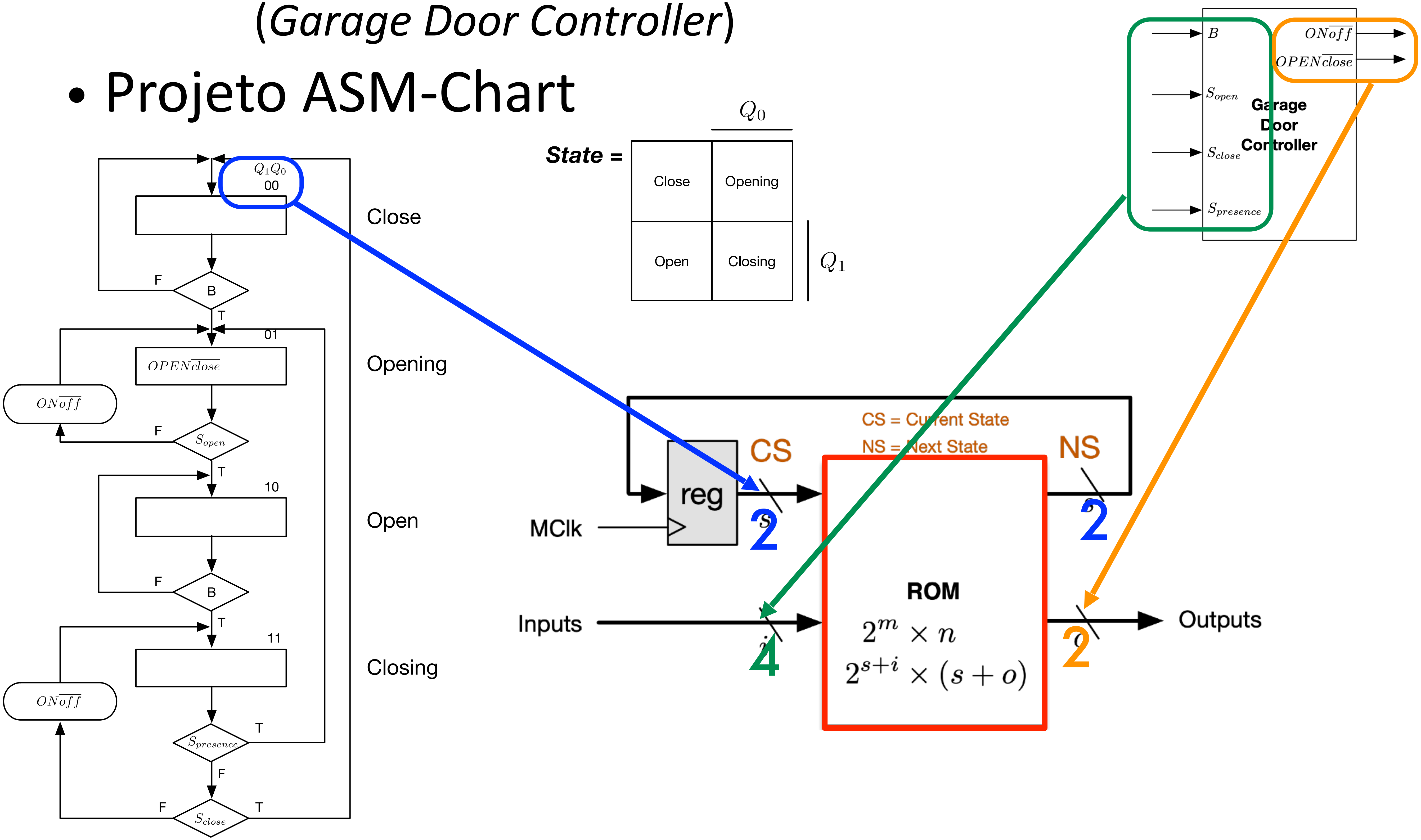




Projeto: Controlo de Portão (3ª versão)

(*Garage Door Controller*)

- Projeto ASM-Chart



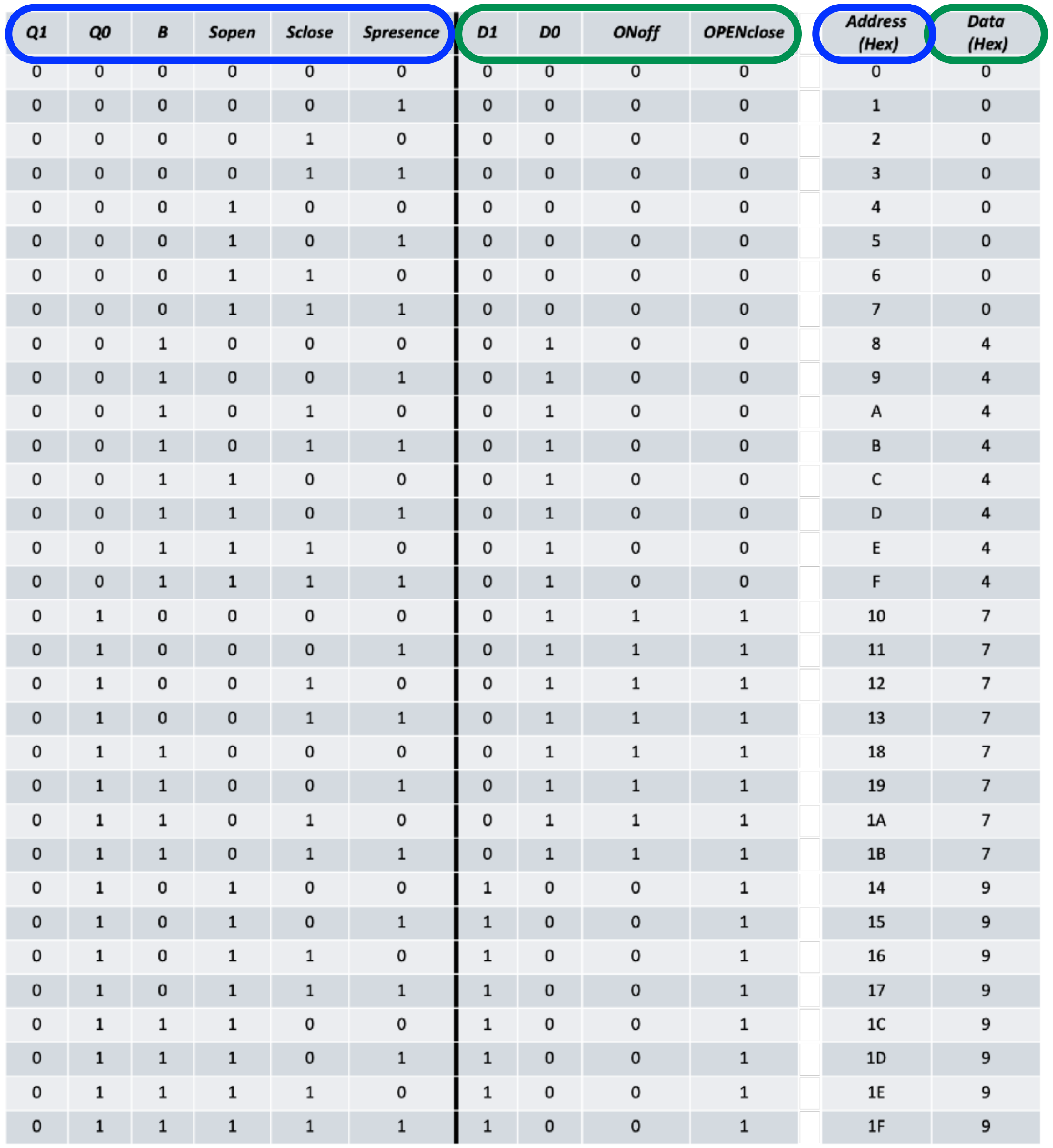
State =

	Q_0	
	Close	Opening
Q_1	Open	Closing

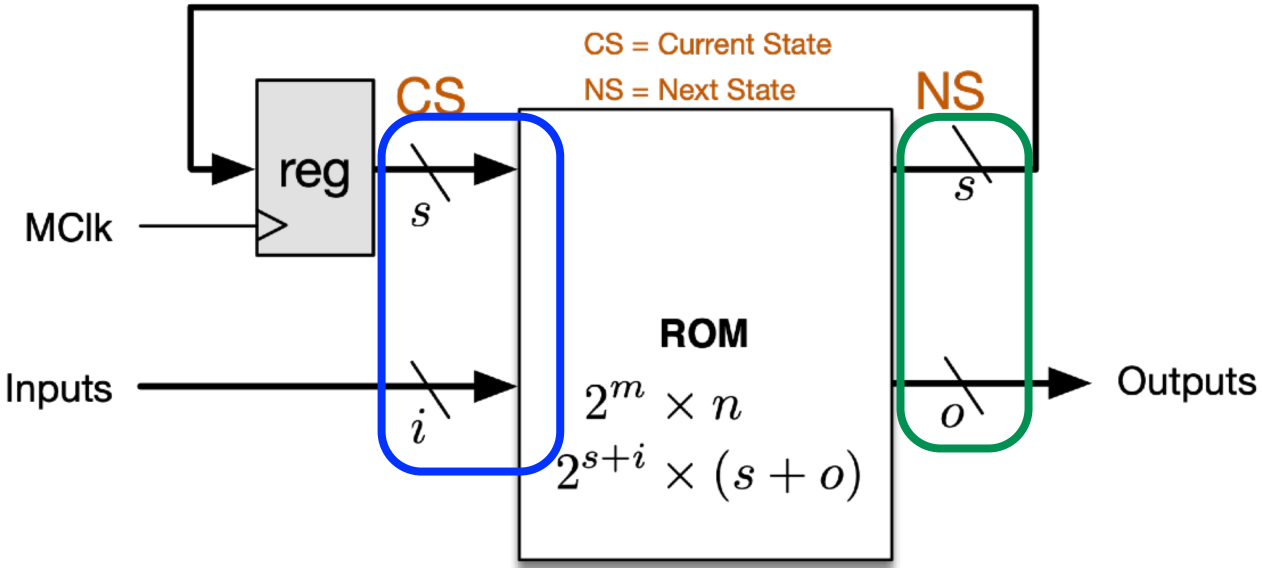
Projeto: Controlo de Portão (3ª versão)

(*Garage Door Controller*)

- Projeto ASM-Chart



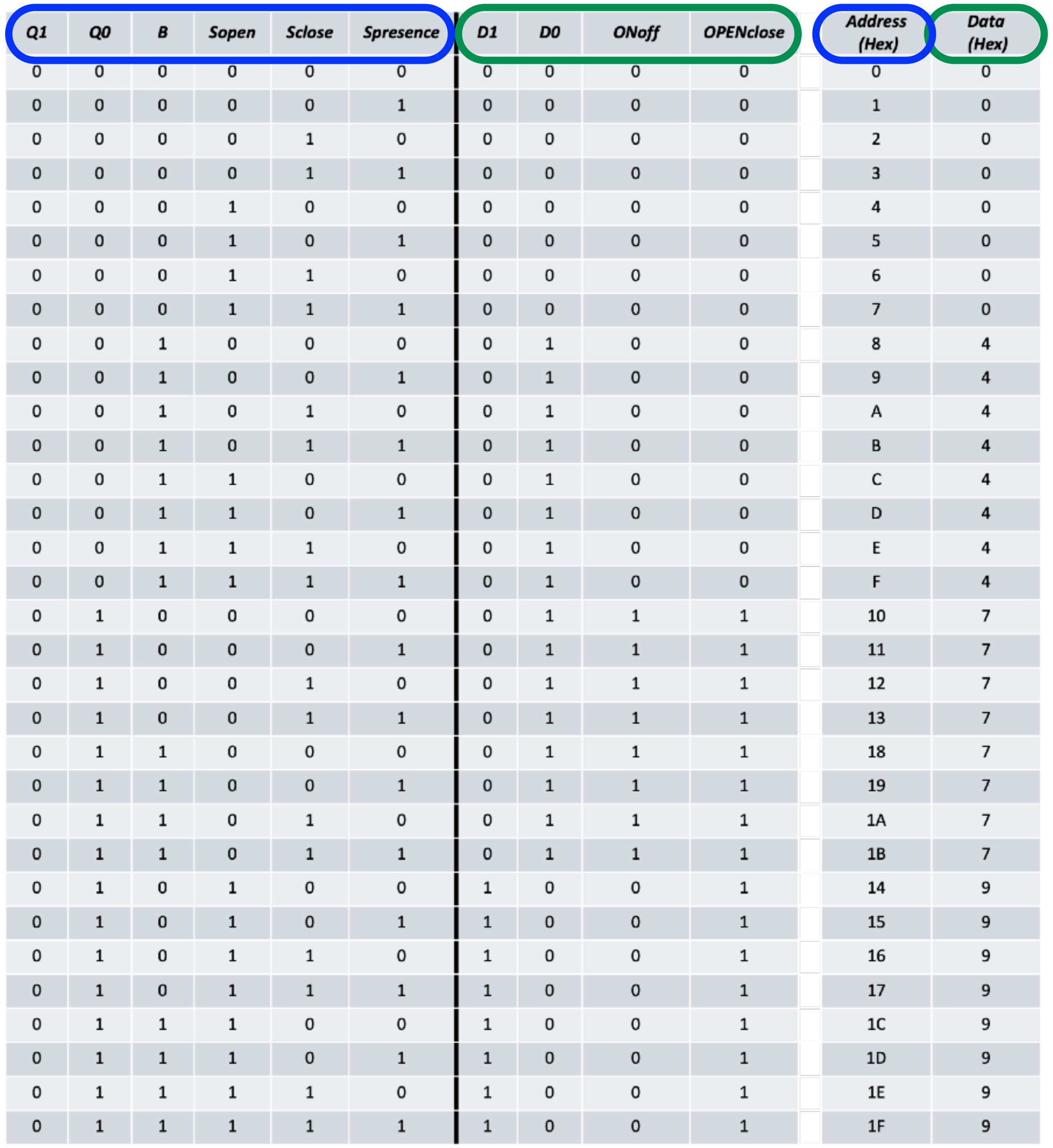
Q1	Q0	B	Sopen	Sclose	Spresence	D1	D0	ONoff	OPENclose		Address (Hex)	Data (Hex)
0	0	0	0	0	0	0	0	0	0		0	0
0	0	0	0	0	1	0	0	0	0		1	0
0	0	0	0	1	0	0	0	0	0		2	0
0	0	0	0	1	1	0	0	0	0		3	0
0	0	0	1	0	0	0	0	0	0		4	0
0	0	0	1	0	1	0	0	0	0		5	0
0	0	0	1	1	0	0	0	0	0		6	0
0	0	0	1	1	1	0	0	0	0		7	0
0	0	1	0	0	0	0	1	0	0		8	4
0	0	1	0	0	1	0	1	0	0		9	4
0	0	1	0	1	0	0	1	0	0		A	4
0	0	1	0	1	1	0	1	0	0		B	4
0	0	1	1	0	0	0	1	0	0		C	4
0	0	1	1	0	1	0	1	0	0		D	4
0	0	1	1	1	0	0	1	0	0		E	4
0	0	1	1	1	1	0	1	0	0		F	4
0	1	0	0	0	0	0	1	1	1		10	7
0	1	0	0	0	1	0	1	1	1		11	7
0	1	0	0	1	0	0	1	1	1		12	7
0	1	0	0	1	1	0	1	1	1		13	7
0	1	1	0	0	0	0	1	1	1		18	7
0	1	1	0	0	1	0	1	1	1		19	7
0	1	1	0	1	0	0	1	1	1		1A	7
0	1	1	0	1	1	0	1	1	1		1B	7
0	1	0	1	0	0	1	0	0	1		14	9
0	1	0	1	0	1	1	0	0	1		15	9
0	1	0	1	1	0	1	0	0	1		16	9
0	1	0	1	1	1	1	0	0	1		17	9
0	1	1	1	0	0	1	0	0	1		1C	9
0	1	1	1	0	1	1	0	0	1		1D	9
0	1	1	1	1	0	1	0	0	1		1E	9
0	1	1	1	1	1	1	0	0	1		1F	9



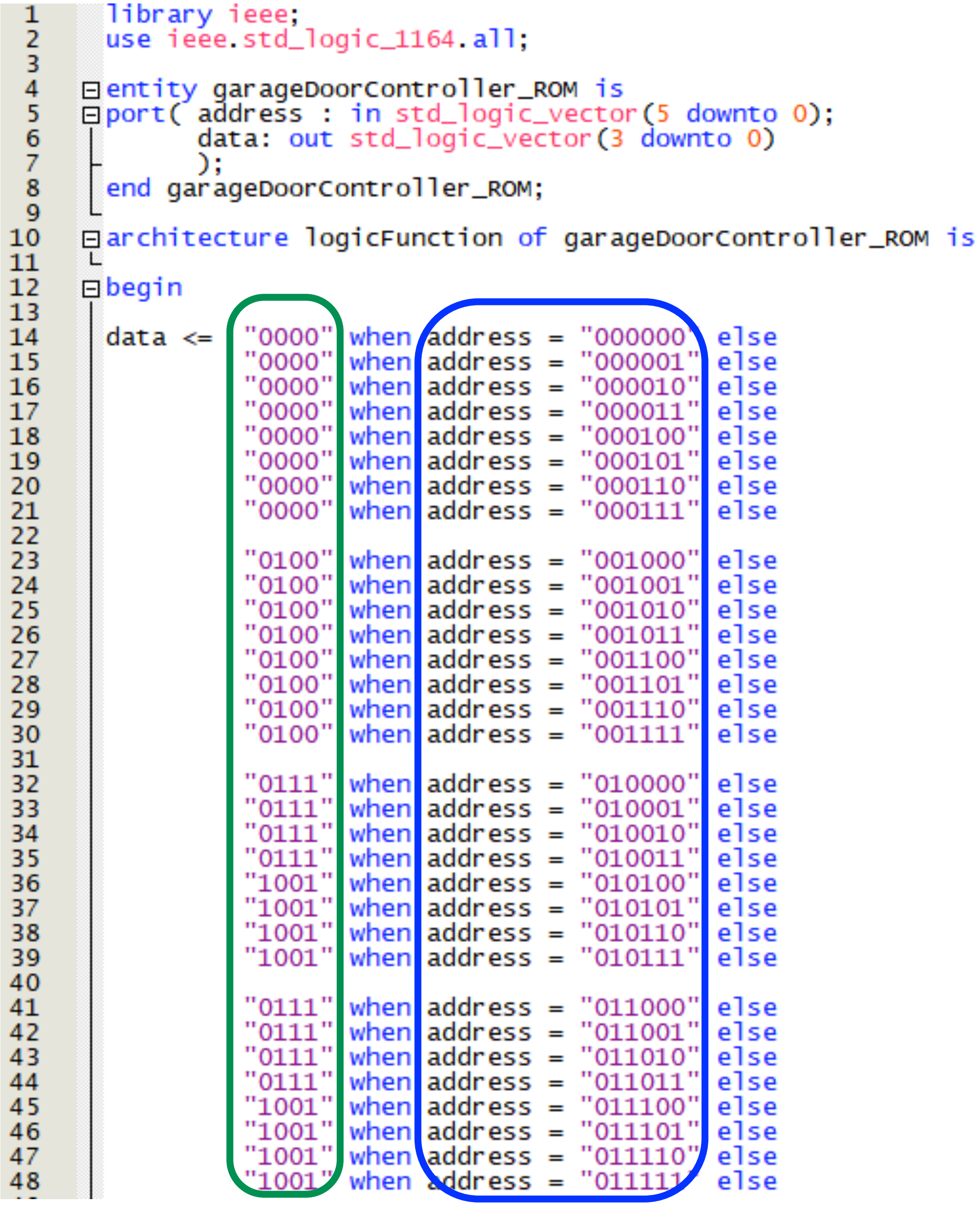
Projeto: Controlo de Portão (3ª versão)

(*Garage Door Controller*)

- Projeto ASM-Chart



Q1	Q0	B	Sopen	Sclose	Spresence	D1	D0	ONoff	OPENclose	Address (Hex)	Data (Hex)
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	1	0	0	0	0	1	0
0	0	0	0	1	0	0	0	0	0	2	0
0	0	0	0	1	1	0	0	0	0	3	0
0	0	0	1	0	0	0	0	0	0	4	0
0	0	0	1	0	1	0	0	0	0	5	0
0	0	0	1	1	0	0	0	0	0	6	0
0	0	0	1	1	1	0	0	0	0	7	0
0	0	1	0	0	0	0	1	0	0	8	4
0	0	1	0	0	1	0	1	0	0	9	4
0	0	1	0	1	0	0	1	0	0	A	4
0	0	1	0	1	1	0	1	0	0	B	4
0	0	1	1	0	0	0	1	0	0	C	4
0	0	1	1	0	1	0	1	0	0	D	4
0	0	1	1	1	0	0	1	0	0	E	4
0	0	1	1	1	1	0	1	0	0	F	4
0	1	0	0	0	0	0	1	1	1	10	7
0	1	0	0	0	1	0	1	1	1	11	7
0	1	0	0	1	0	0	1	1	1	12	7
0	1	0	0	1	1	0	1	1	1	13	7
0	1	1	0	0	0	0	1	1	1	18	7
0	1	1	0	0	1	0	1	1	1	19	7
0	1	1	0	1	0	0	1	1	1	1A	7
0	1	1	0	1	1	0	1	1	1	1B	7
0	1	0	1	0	0	1	0	0	1	14	9
0	1	0	1	0	1	1	0	0	1	15	9
0	1	0	1	1	0	1	0	0	1	16	9
0	1	0	1	1	1	1	0	0	1	17	9
0	1	1	1	0	0	1	0	0	1	1C	9
0	1	1	1	0	1	1	0	0	1	1D	9
0	1	1	1	1	0	1	0	0	1	1E	9
0	1	1	1	1	1	1	0	0	1	1F	9

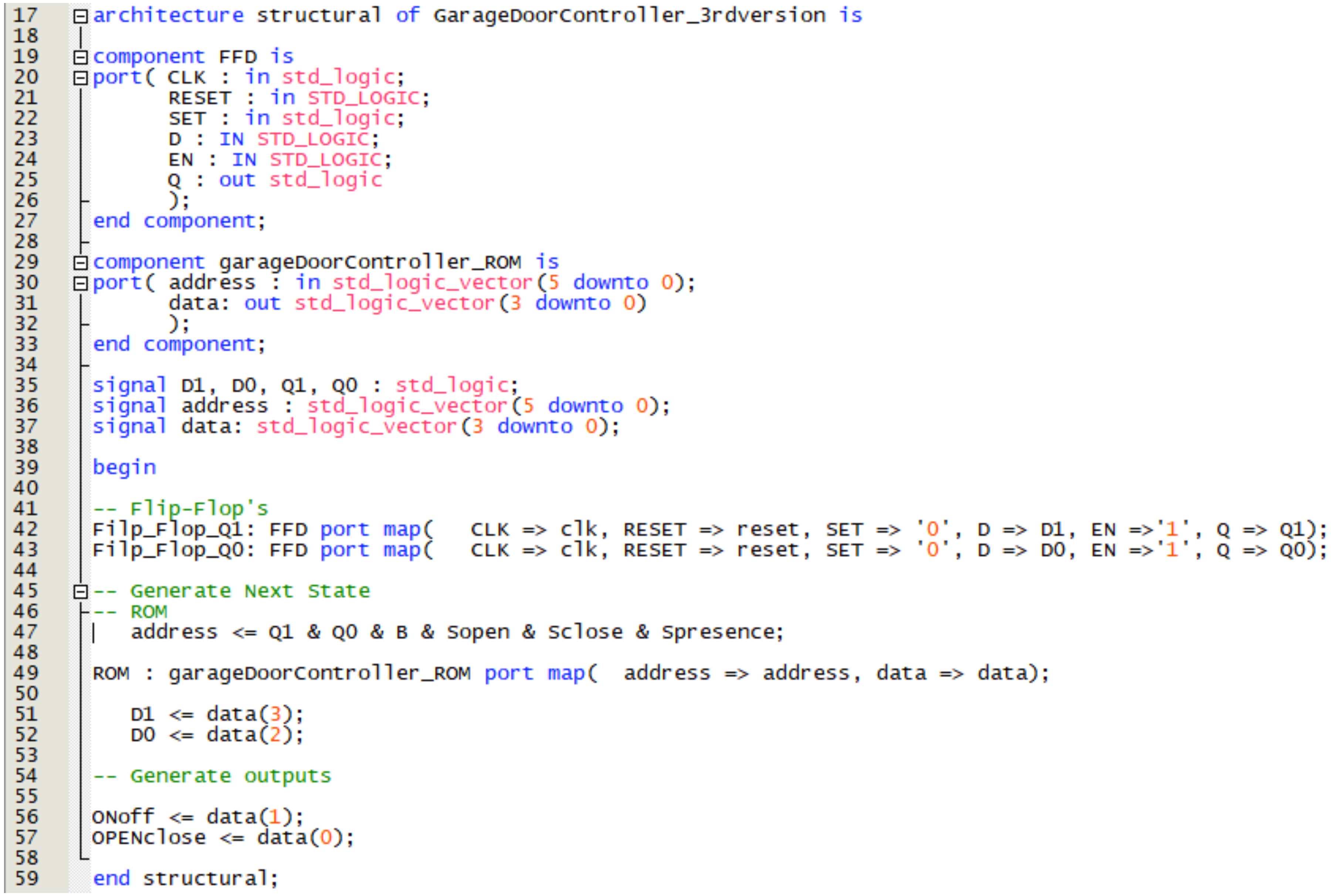


```vhdl
library ieee;
use ieee.std_logic_1164.all;

entity garageDoorController_ROM is
port( address : in std_logic_vector(5 downto 0);
          data: out std_logic_vector(3 downto 0)
          );
end garageDoorController_ROM;

architecture logicFunction of garageDoorController_ROM is

begin

data <=    "0000" when address = "000000" else
           "0000" when address = "000001" else
           "0000" when address = "000010" else
           "0000" when address = "000011" else
           "0000" when address = "000100" else
           "0000" when address = "000101" else
           "0000" when address = "000110" else
           "0000" when address = "000111" else

           "0100" when address = "001000" else
           "0100" when address = "001001" else
           "0100" when address = "001010" else
           "0100" when address = "001011" else
           "0100" when address = "001100" else
           "0100" when address = "001101" else
           "0100" when address = "001110" else
           "0100" when address = "001111" else

           "0111" when address = "010000" else
           "0111" when address = "010001" else
           "0111" when address = "010010" else
           "0111" when address = "010011" else
           "1001" when address = "010100" else
           "1001" when address = "010101" else
           "1001" when address = "010110" else
           "1001" when address = "010111" else

           "0111" when address = "011000" else
           "0111" when address = "011001" else
           "0111" when address = "011010" else
           "0111" when address = "011011" else
           "1001" when address = "011100" else
           "1001" when address = "011101" else
           "1001" when address = "011110" else
           "1001" when address = "011111" else
```

Projeto: Controlo de Portão (3ª versão)

(*Garage Door Controller*)

- Projeto ASM-Chart

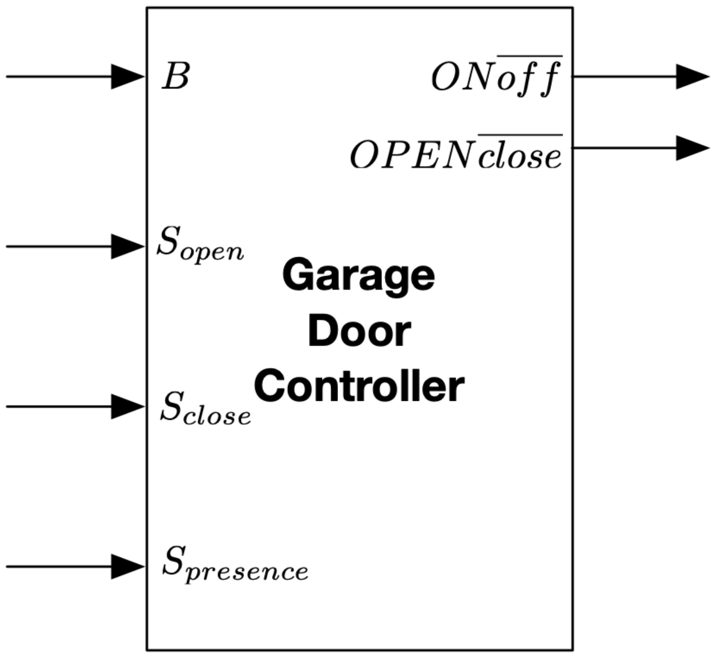


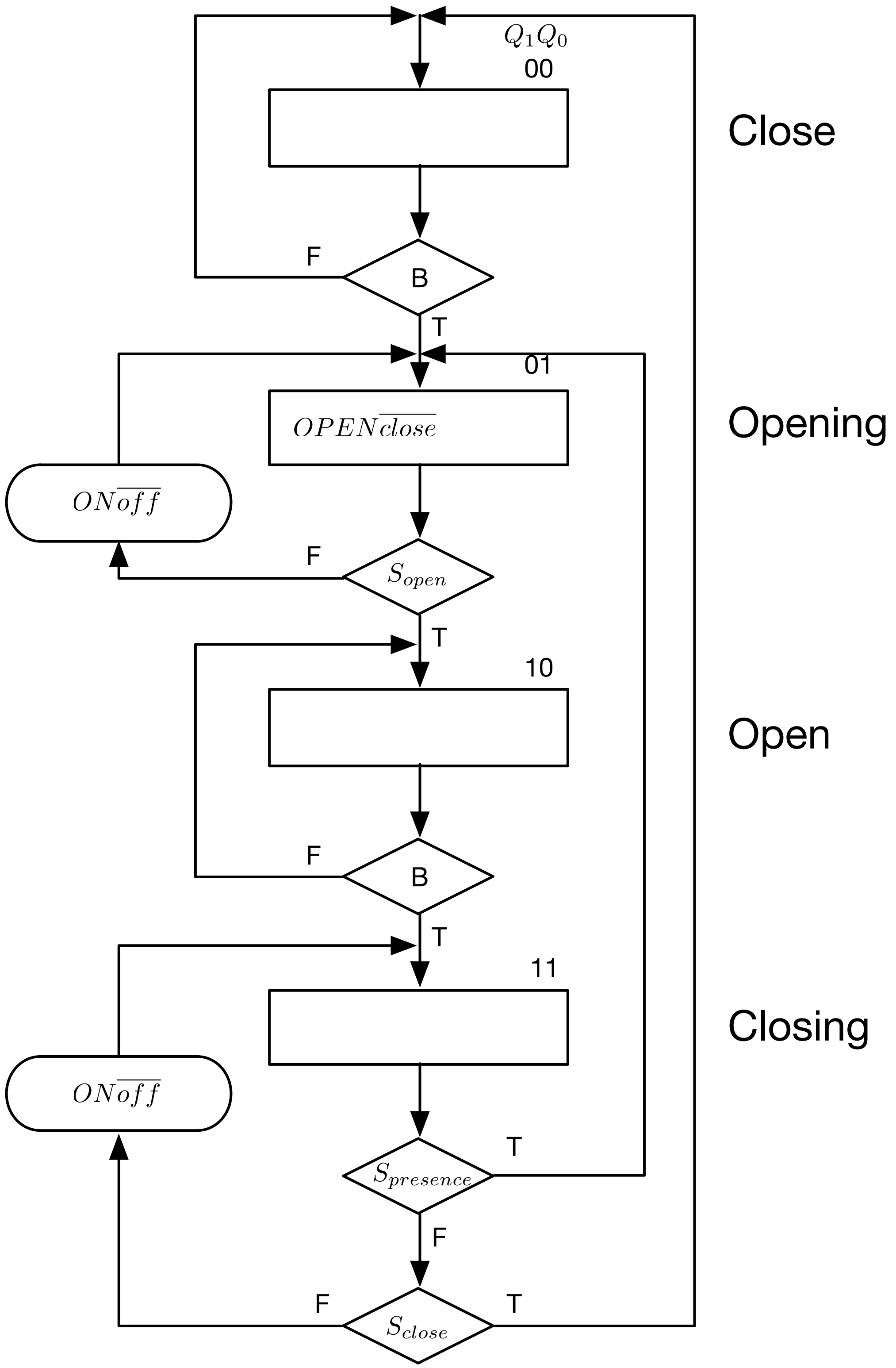
```
architecture structural of GarageDoorController_3rdversion is

component FFD is
port( CLK : in std_logic;
      RESET : in STD_LOGIC;
      SET : in std_logic;
      D : IN STD_LOGIC;
      EN : IN STD_LOGIC;
      Q : out std_logic
      );
end component;

component garageDoorController_ROM is
port( address : in std_logic_vector(5 downto 0);
      data: out std_logic_vector(3 downto 0)
      );
end component;

signal D1, D0, Q1, Q0 : std_logic;
signal address : std_logic_vector(5 downto 0);
signal data: std_logic_vector(3 downto 0);

begin

-- Flip-Flop's
Filp_Flop_Q1: FFD port map(   CLK => clk, RESET => reset, SET => '0', D => D1, EN =>'1', Q => Q1);
Filp_Flop_Q0: FFD port map(   CLK => clk, RESET => reset, SET => '0', D => D0, EN =>'1', Q => Q0);

-- Generate Next State
-- ROM
   address <= Q1 & Q0 & B & Sopen & Sclose & Spresence;

ROM : garageDoorController_ROM port map(  address => address, data => data);

   D1 <= data(3);
   D0 <= data(2);

-- Generate outputs

ONoff  <= data(1);
OPENClose  <= data(0);

end structural;
```

Projeto: Controlo de Portão (4ª versão)

(*Garage Door Controller*)

- Arquitetura do sistema

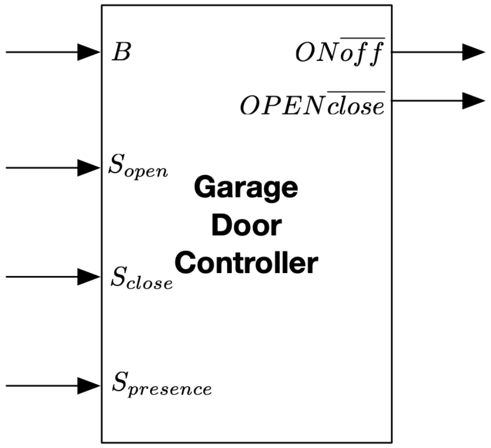


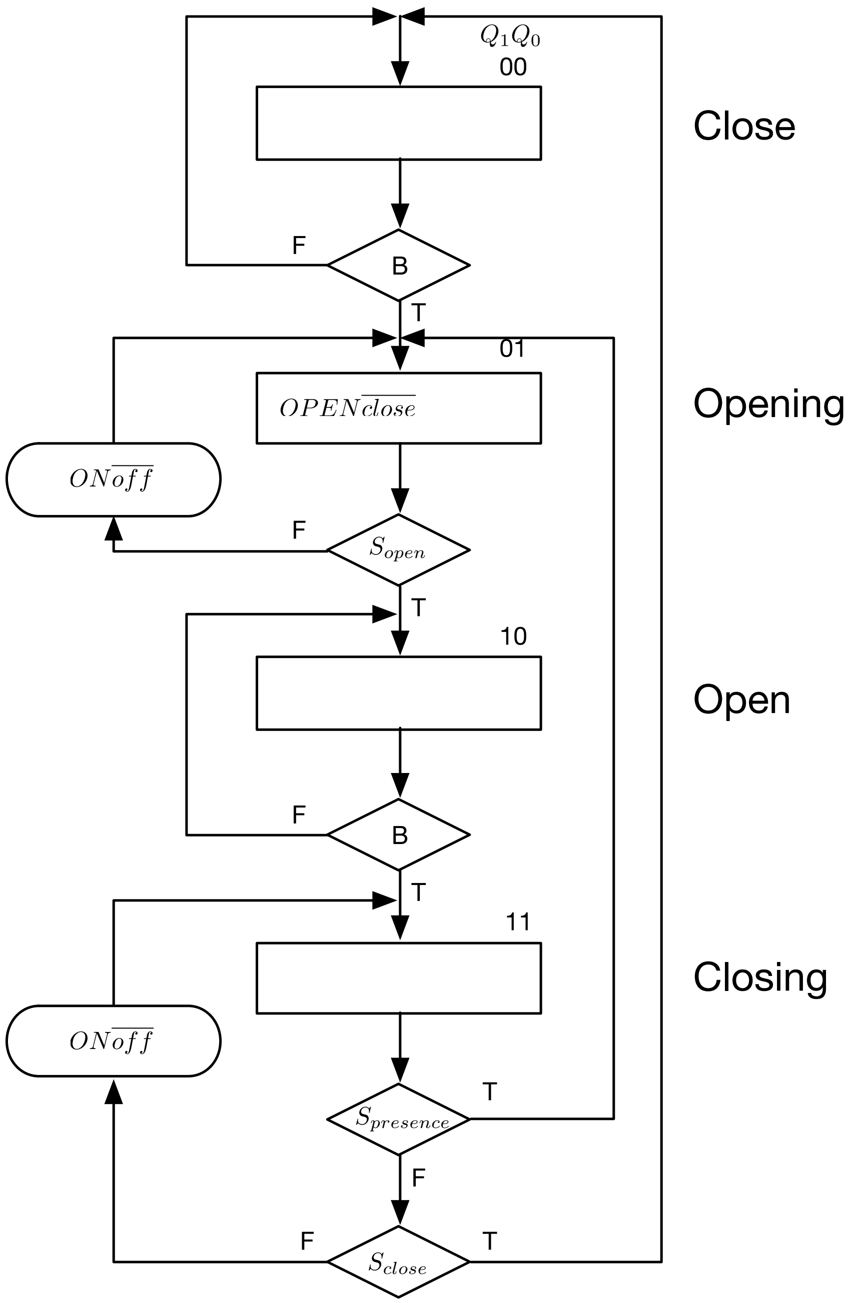


Projeto: Controlo de Portão (4ª versão)

(*Garage Door Controller*)

- Projeto ASM-Chart



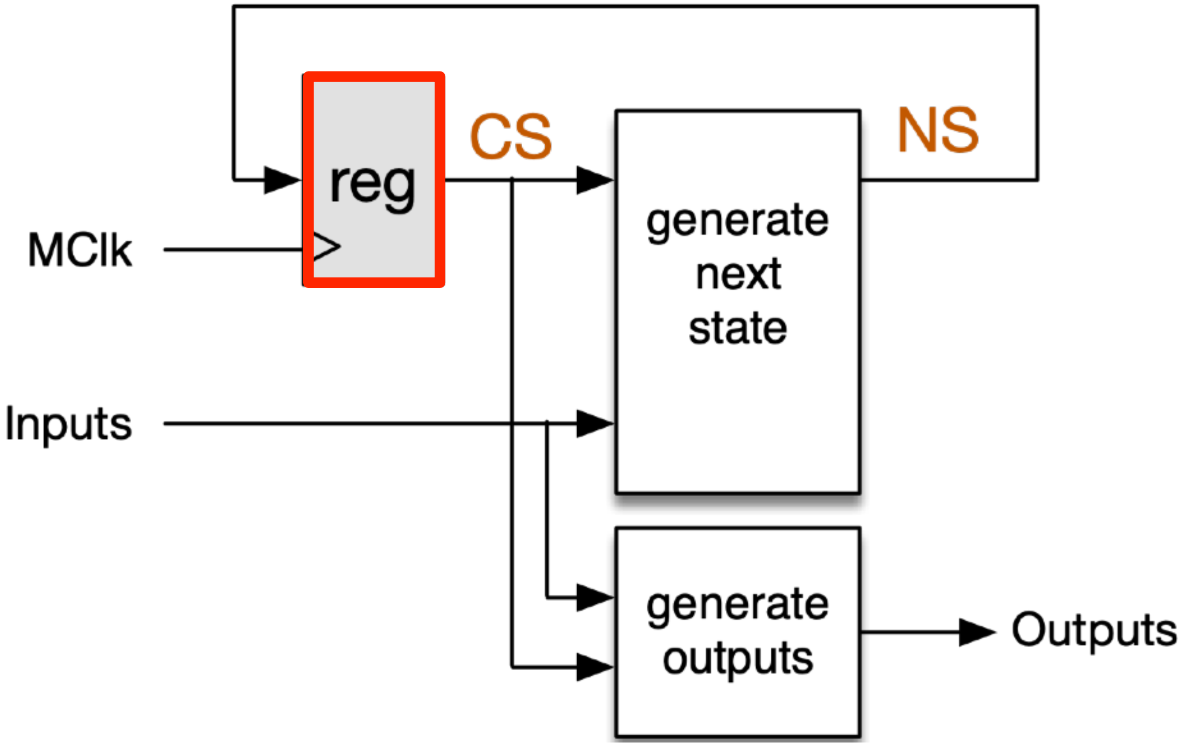


State =

	Q_0	
Close	Opening	
Open	Closing	Q_1

CS = Current State

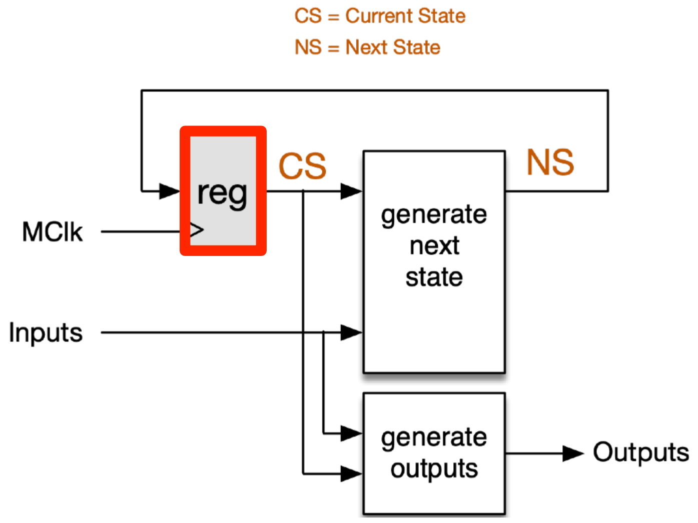
NS = Next State

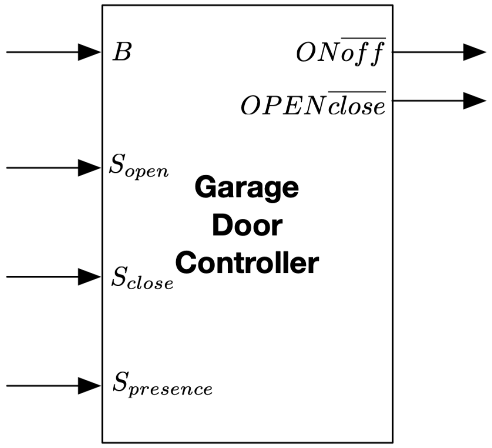


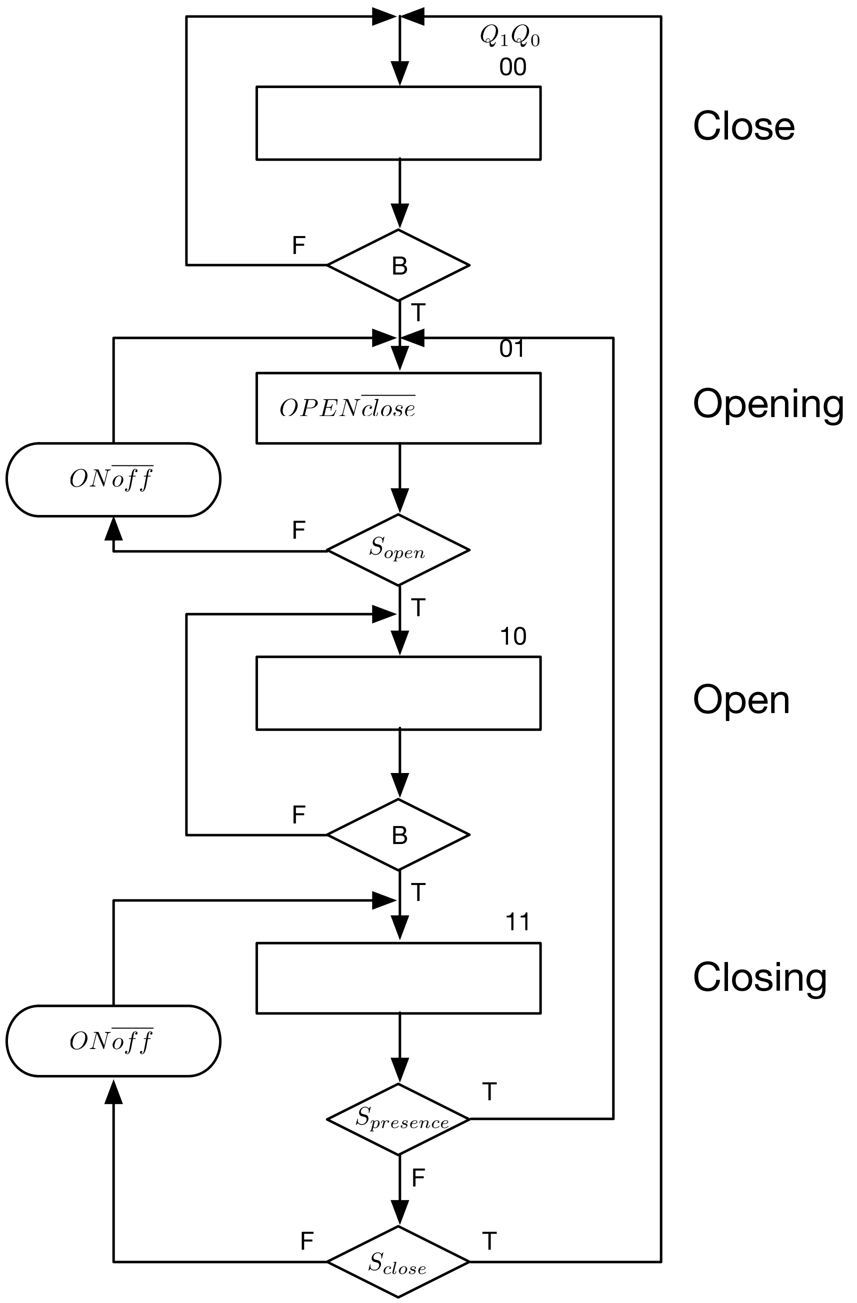
Projeto: Controlo de Portão (4ª versão)

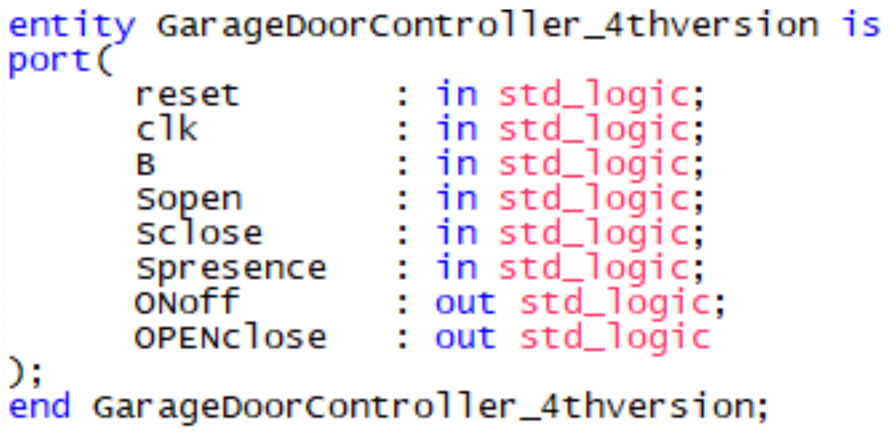
(*Garage Door Controller*)

- Projeto ASM-Chart

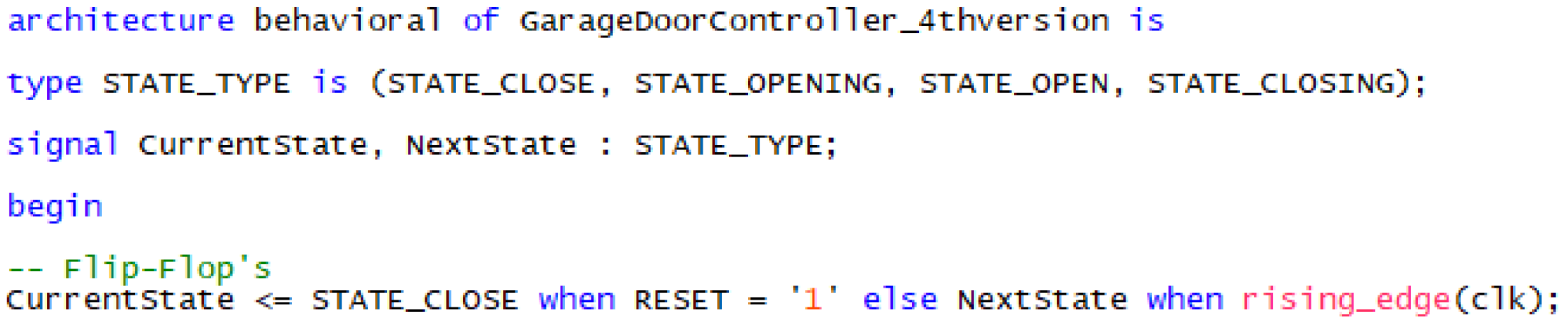








```vhdl
entity GarageDoorController_4thversion is
port(
       reset        : in std_logic;
       clk          : in std_logic;
       B            : in std_logic;
       Sopen        : in std_logic;
       Sclose       : in std_logic;
       Spresence    : in std_logic;
       ONoff        : out std_logic;
       OPENClose    : out std_logic
);
end GarageDoorController_4thversion;
```

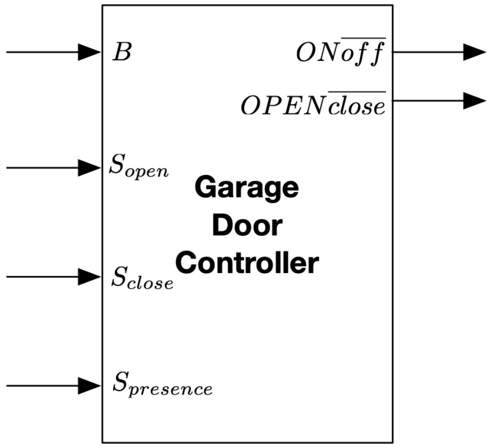


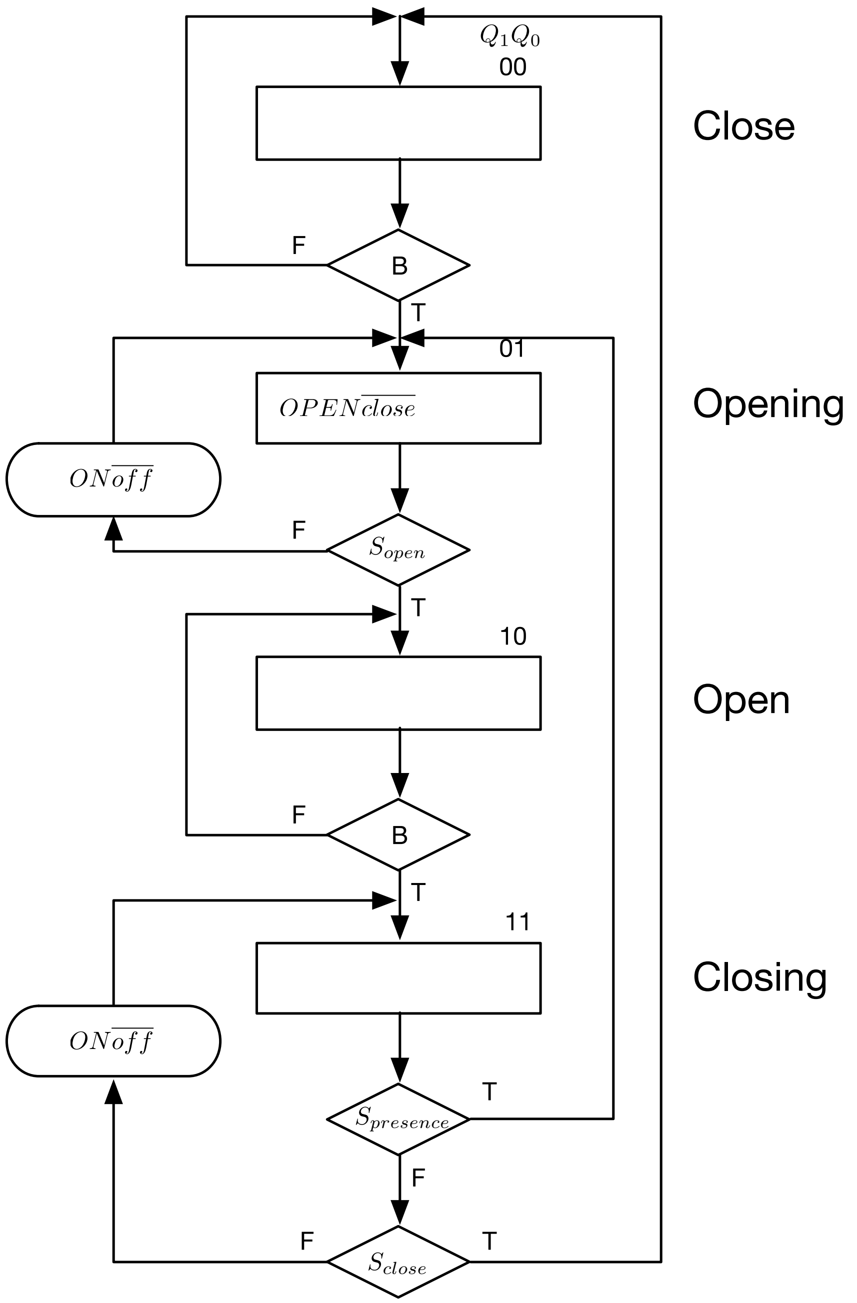
```vhdl
architecture behavioral of GarageDoorController_4thversion is

type STATE_TYPE is (STATE_CLOSE, STATE_OPENING, STATE_OPEN, STATE_CLOSING);

signal CurrentState, NextState : STATE_TYPE;

begin

-- Flip-Flop's
CurrentState <= STATE_CLOSE when RESET = '1' else NextState when rising_edge(clk);
```

Projeto: Controlo de Portão (4ª versão)

(*Garage Door Controller*)

- Projeto ASM-Chart



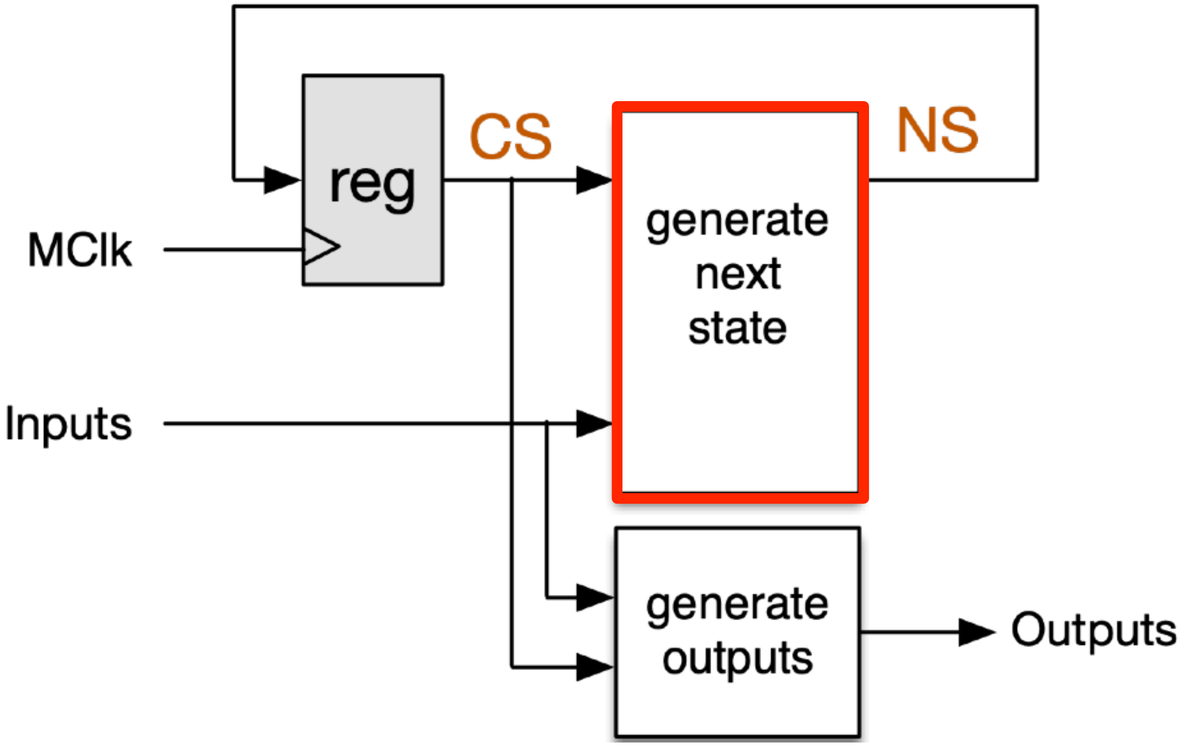


State =

		Q_0
	Close	Opening
Q_1	Open	Closing

CS = Current State

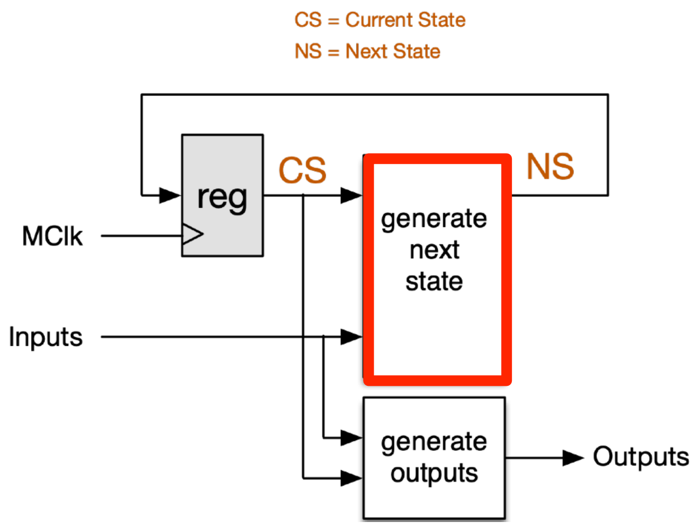
NS = Next State

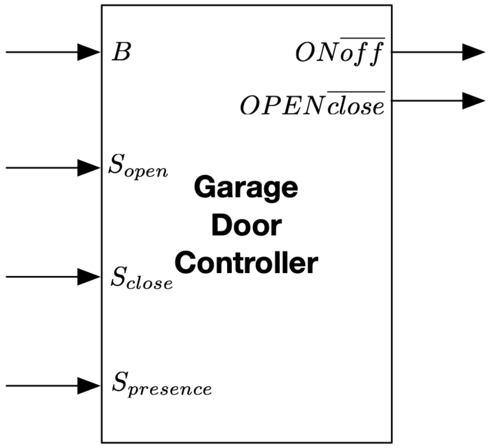


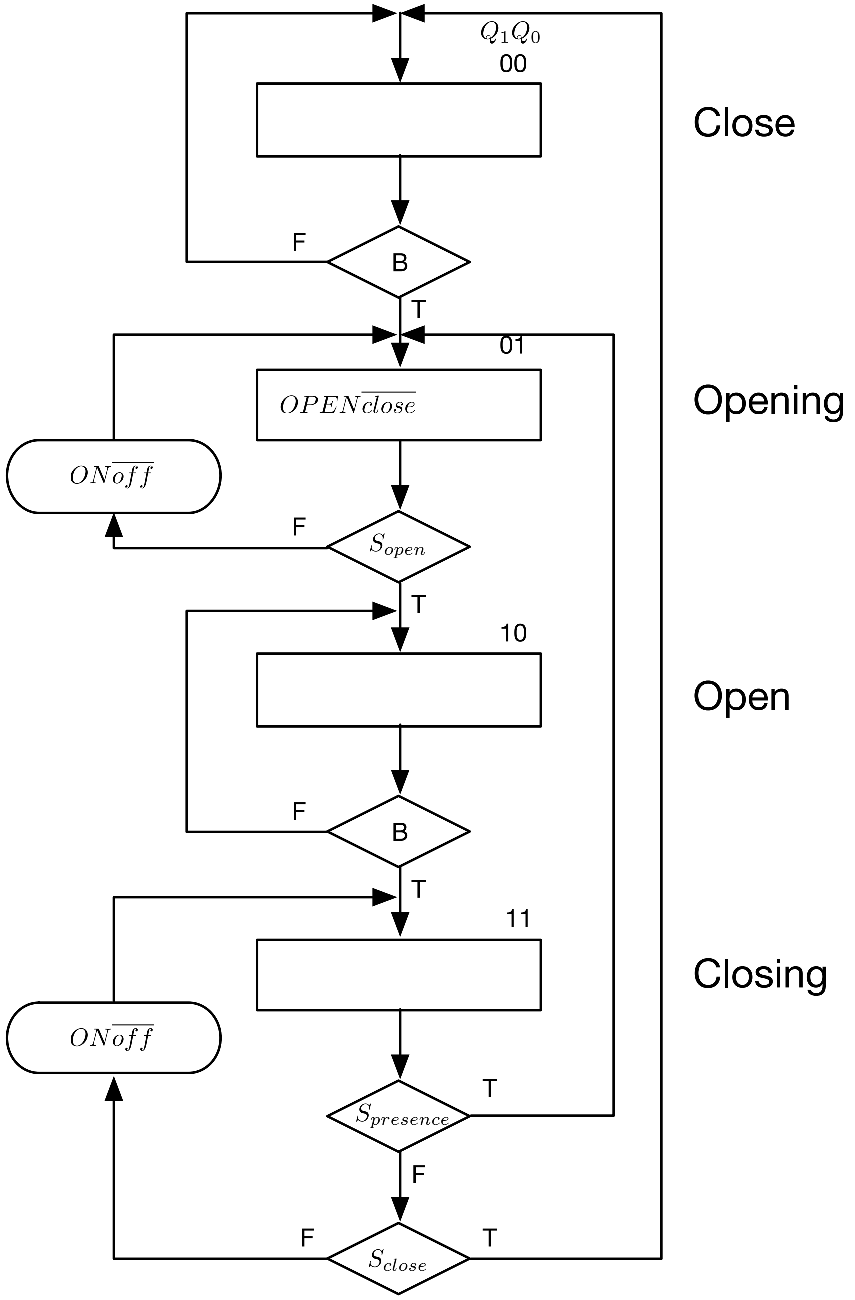
Projeto: Controlo de Portão (4ª versão)

(*Garage Door Controller*)

- Projeto ASM-Chart





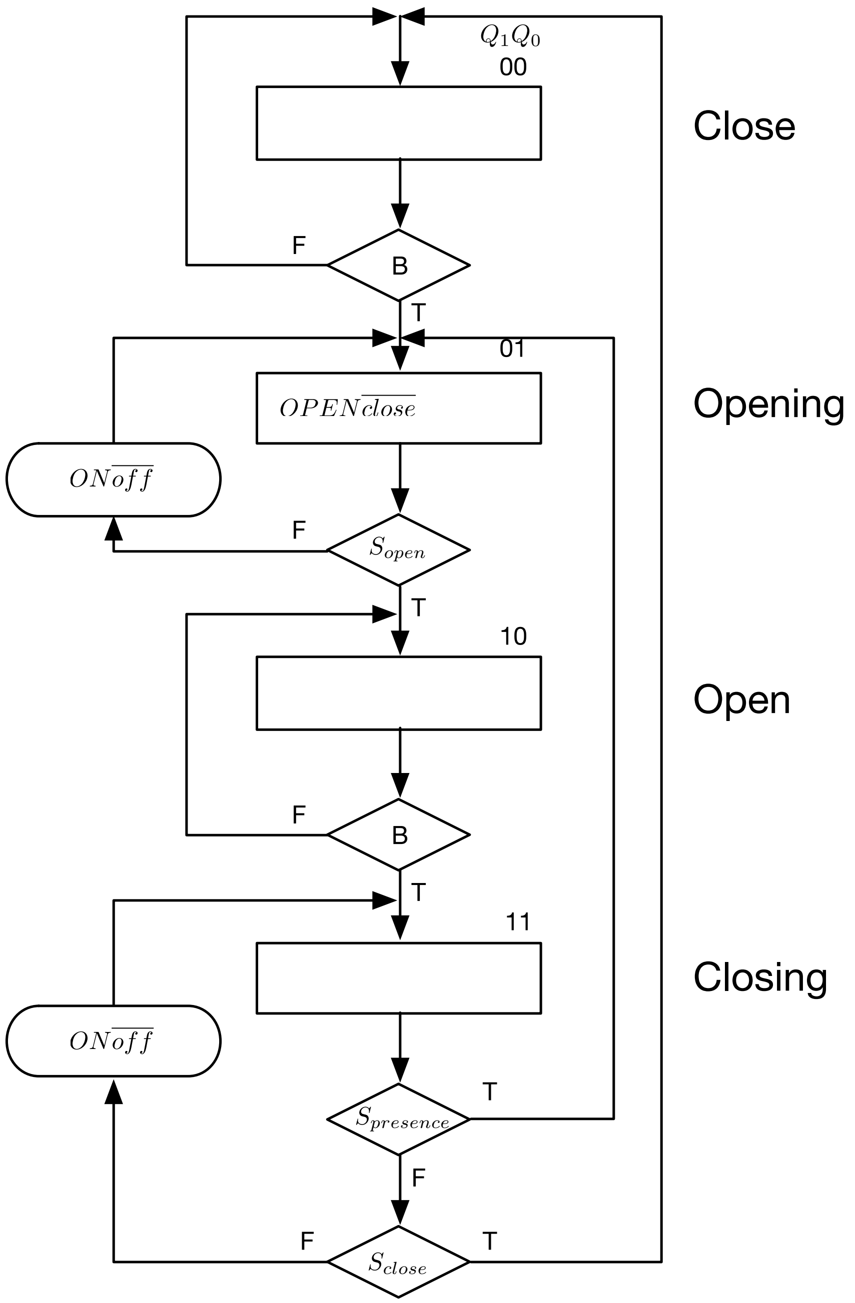


```
-- Generate Next State
GenerateNextState:
process (CurrentState, B, Sopen, Sclose, Spresence)
   begin
      case CurrentState is
         when STATE_CLOSE      => if (B = '1') then
                                     NextState <= STATE_OPENING;
                                  else
                                     NextState <= STATE_CLOSE;
                                  end if;

         when STATE_OPENING    => if (Sopen = '1') then
                                     NextState <= STATE_OPEN;
                                  else
                                     NextState <= STATE_OPENING;
                                  end if;

         when STATE_OPEN       => if (B = '1') then
                                     NextState <= STATE_CLOSING;
                                  else
                                     NextState <= STATE_OPEN;
                                  end if;

         when STATE_CLOSING    => if (Spresence ='1') then
                                     NextState <= STATE_OPENING;
                                  elsif (Spresence ='0' and Sclose = '1') then
                                     NextState <= STATE_CLOSE;
                                  else
                                     NextState <= STATE_CLOSING;
                                  end if;

      end case;
   end process;
```

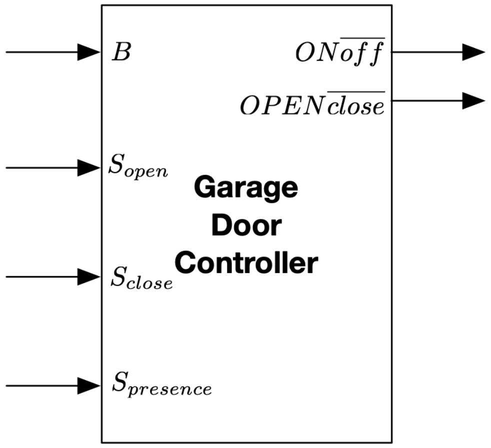
Projeto: Controlo de Portão (4ª versão)

(*Garage Door Controller*)

- Projeto ASM-Chart

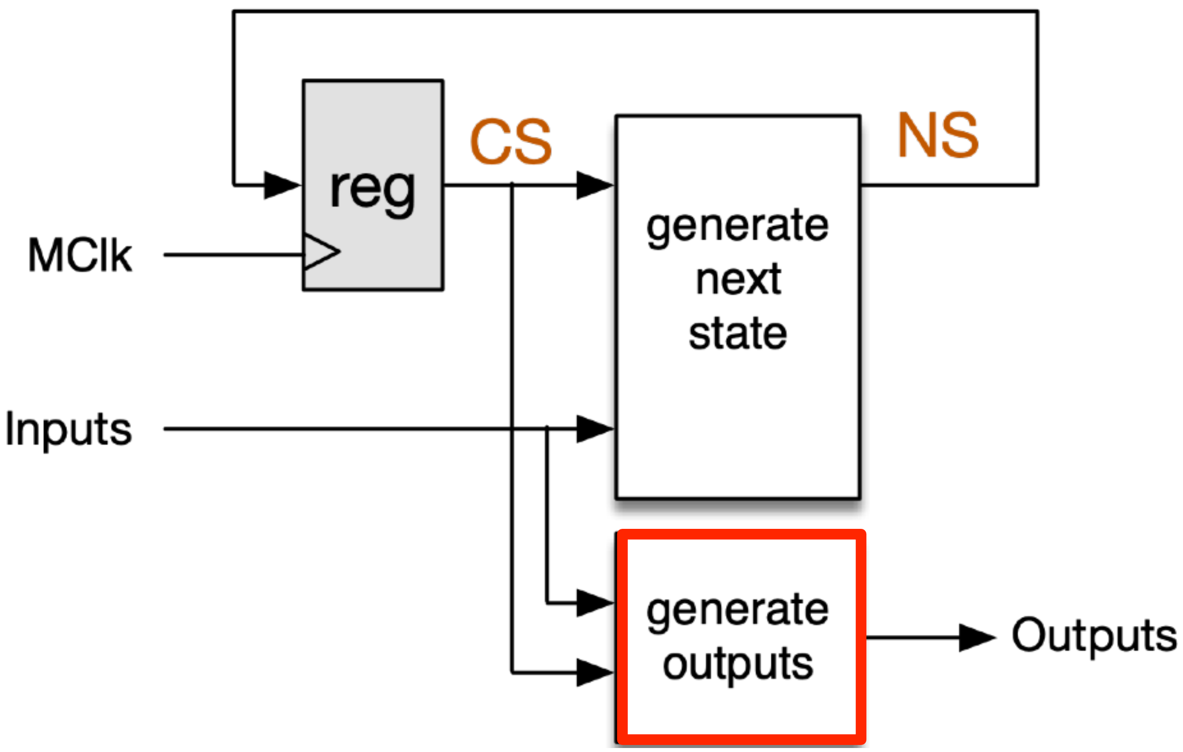


State =		Q_0
	Close	Opening
Q_1	Open	Closing



CS = Current State

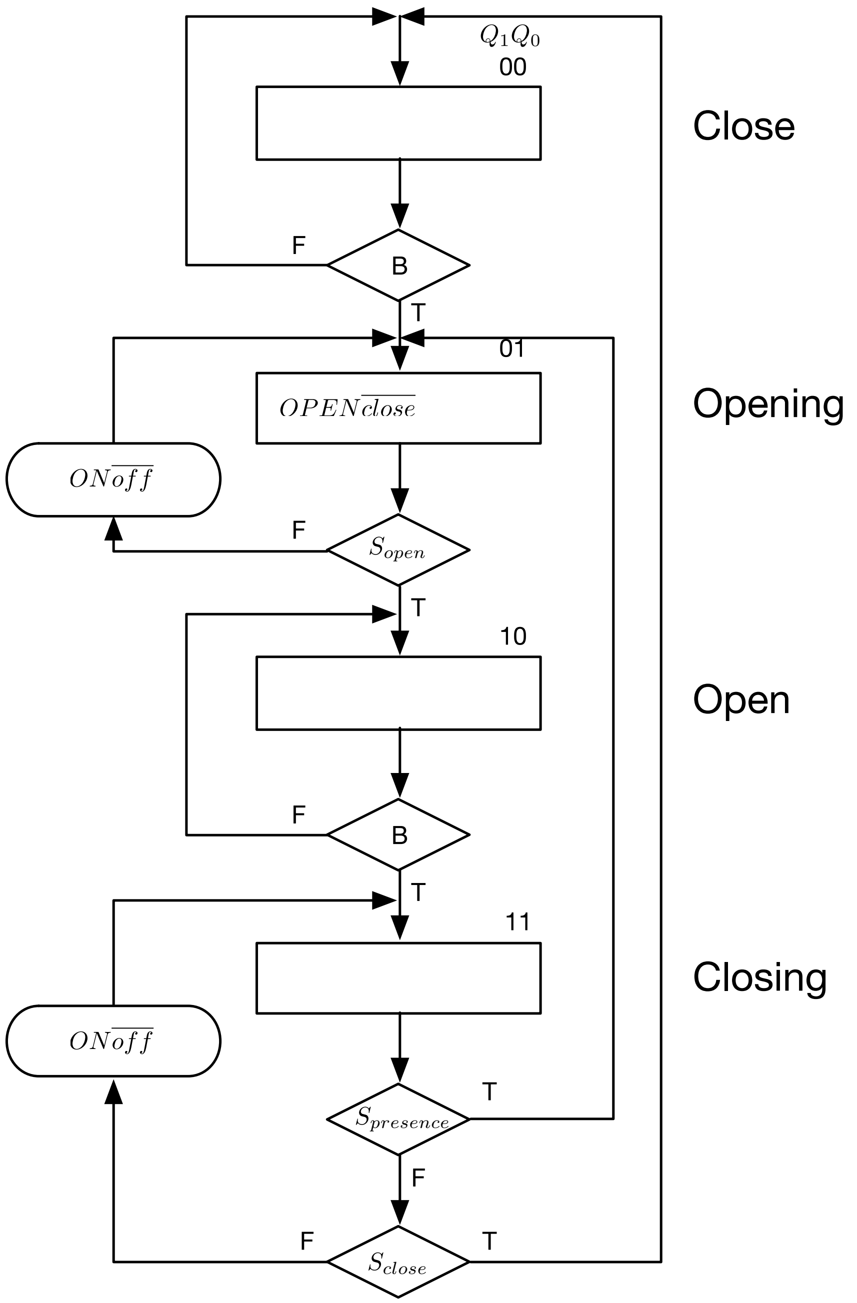
NS = Next State

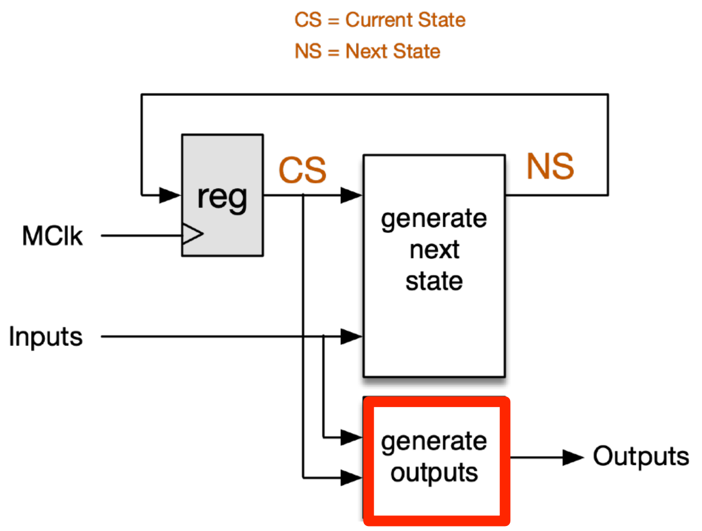


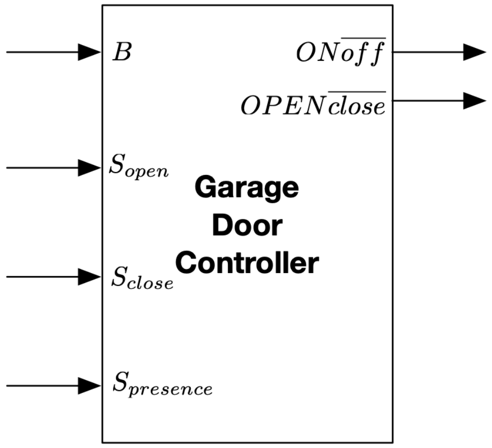
Projeto: Controlo de Portão (4ª versão)

(*Garage Door Controller*)

- Projeto ASM-Chart





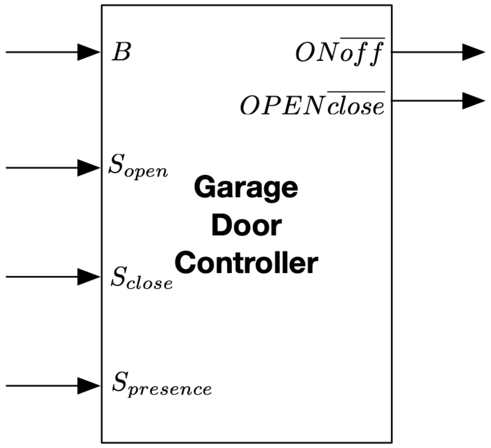


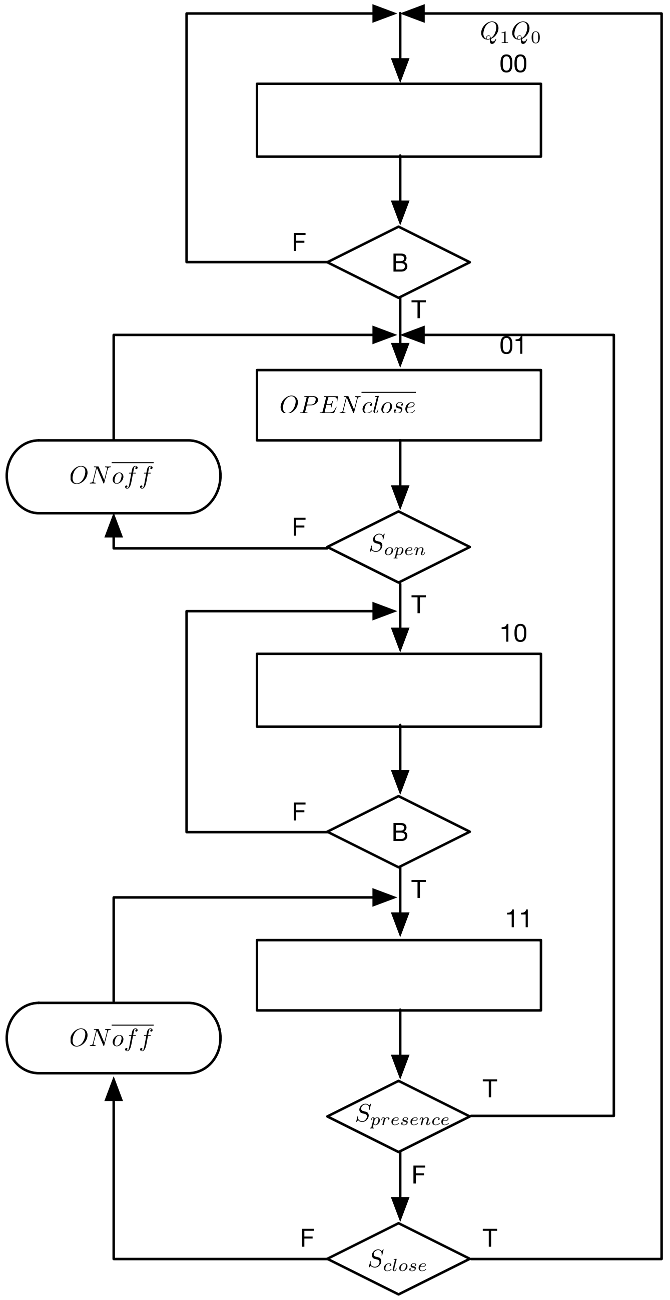
```
-- Generate outputs
ONoff <= '1' when ( (CurrentState = STATE_OPENING and Sopen ='0')
               or (CurrentState = STATE_CLOSING and Spresence ='0' and Sclose = '0'))
               else '0';

OPENclose <= '1' when (CurrentState = STATE_OPENING) else '0';
```

Projeto: Controlo de Portão (4ª versão)

(*Garage Door Controller*)

- Projeto ASM-Chart





Close

Opening

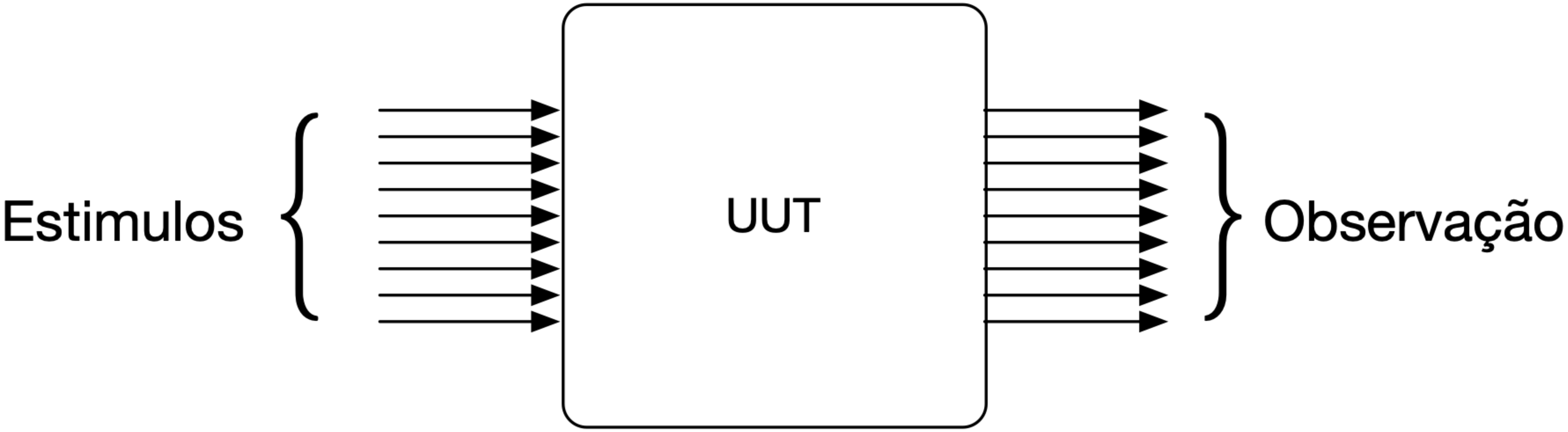
Open

Closing

```
architecture behavioral of GarageDoorController_4thversion is

type STATE_TYPE is (STATE_CLOSE, STATE_OPENING, STATE_OPEN, STATE_CLOSING);

signal CurrentState, NextState : STATE_TYPE;

begin

-- Flip-Flop's
CurrentState <= STATE_CLOSE when RESET = '1' else NextState when rising_edge(clk);

-- Generate Next State
GenerateNextState:
process (CurrentState, B, Sopen, Sclose, Spresence)
   begin
      case CurrentState is
         when STATE_CLOSE      => if (B = '1') then
                                     NextState <= STATE_OPENING;
                                  else
                                     NextState <= STATE_CLOSE;
                                  end if;

         when STATE_OPENING    => if (Sopen = '1') then
                                     NextState <= STATE_OPEN;
                                  else
                                     NextState <= STATE_OPENING;
                                  end if;

         when STATE_OPEN       => if (B = '1') then
                                     NextState <= STATE_CLOSING;
                                  else
                                     NextState <= STATE_OPEN;
                                  end if;

         when STATE_CLOSING    => if (Spresence ='1') then
                                     NextState <= STATE_OPENING;
                                  elsif (Spresence ='0' and Sclose = '1') then
                                     NextState <= STATE_CLOSE;
                                  else
                                     NextState <= STATE_CLOSING;
                                  end if;
      end case;
   end process;

-- Generate outputs
ONoff  <= '1' when ( (CurrentState = STATE_OPENING and Sopen ='0')
               or (CurrentState = STATE_CLOSING and Spresence ='0' and Sclose = '0'))
               else '0';

OPENclose <= '1' when (CurrentState = STATE_OPENING) else '0';

end behavioral;
```

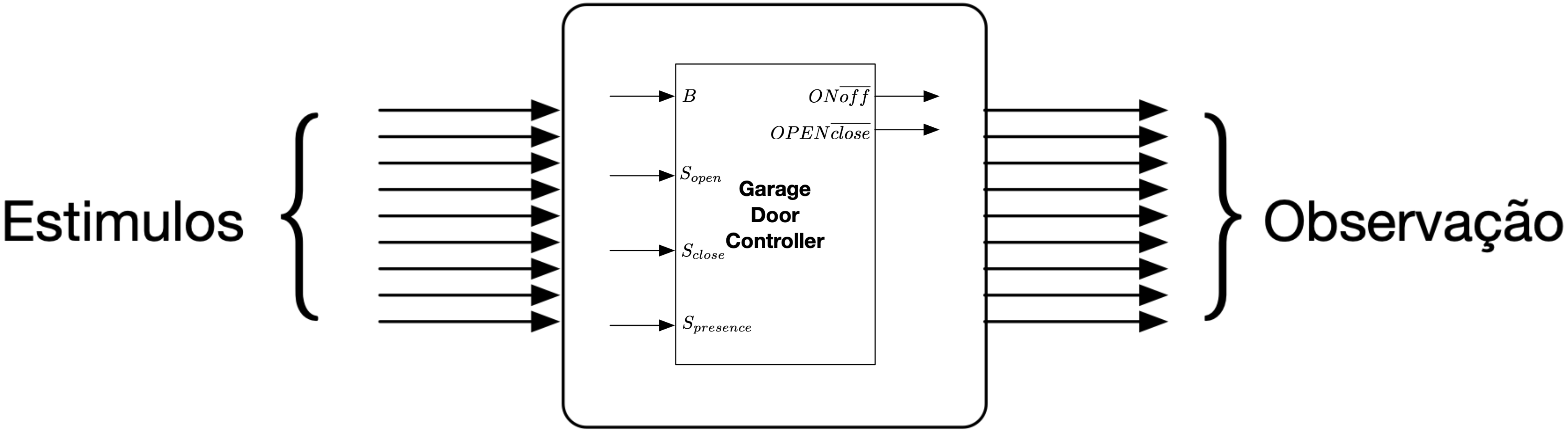
Teste / Validação

- *Unit Under Test*

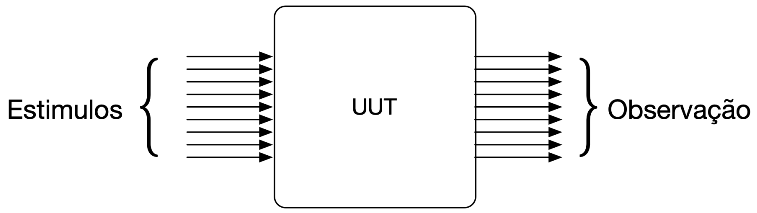


Teste / Validação

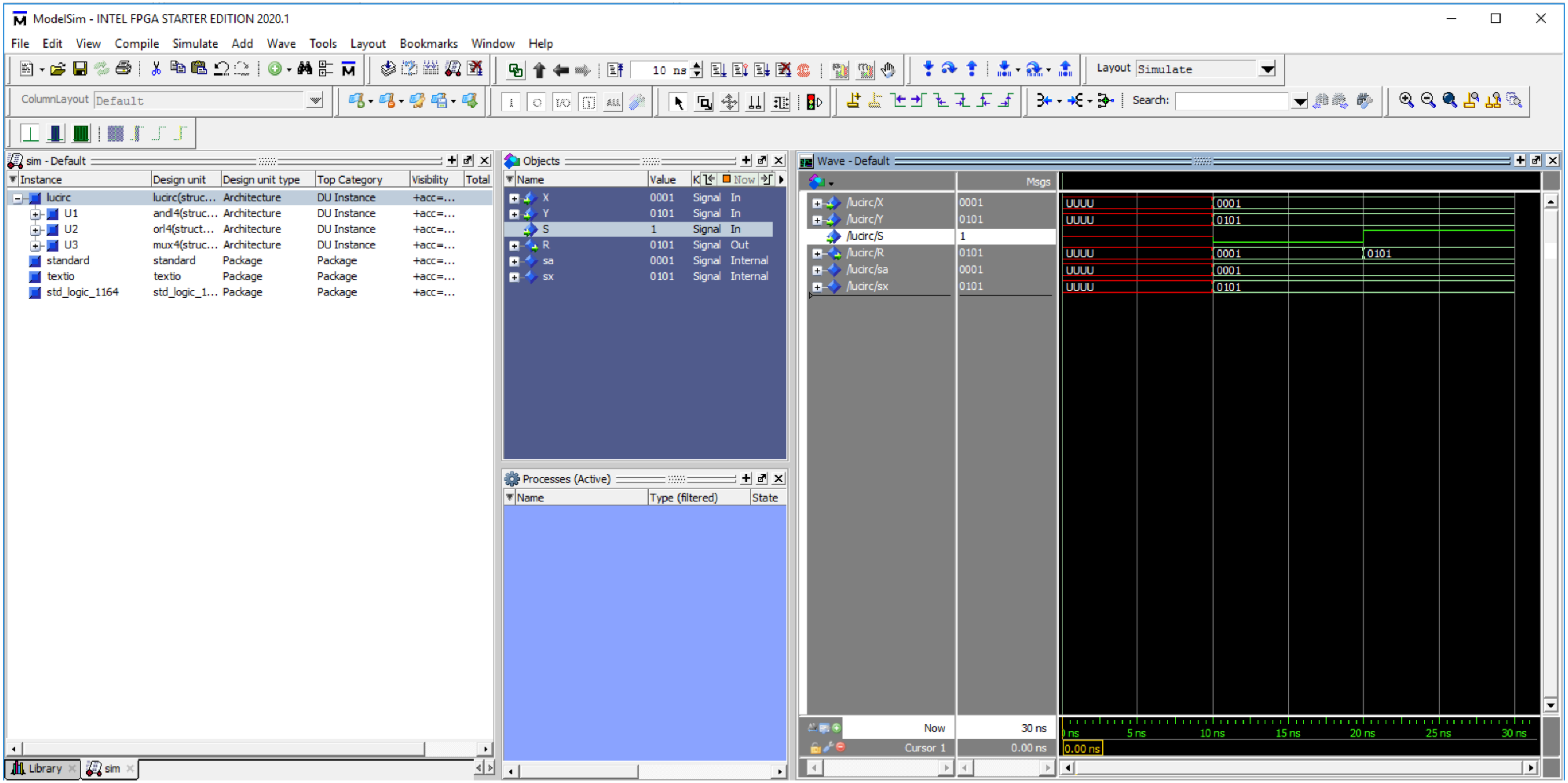
- *Unit Under Test*



Teste / Validação



- *Unit Under Test*



Instance	Design unit	Design unit type	Top Category	Visibility	Total
lucirc	lucirc(struc...	Architecture	DU Instance	+acc=...	
U1	andl4(struc...	Architecture	DU Instance	+acc=...	
U2	orl4(struct...	Architecture	DU Instance	+acc=...	
U3	mux4(struc...	Architecture	DU Instance	+acc=...	
standard	standard	Package	Package	+acc=...	
textio	textio	Package	Package	+acc=...	
std_logic_1164	std_logic_1...	Package	Package	+acc=...	

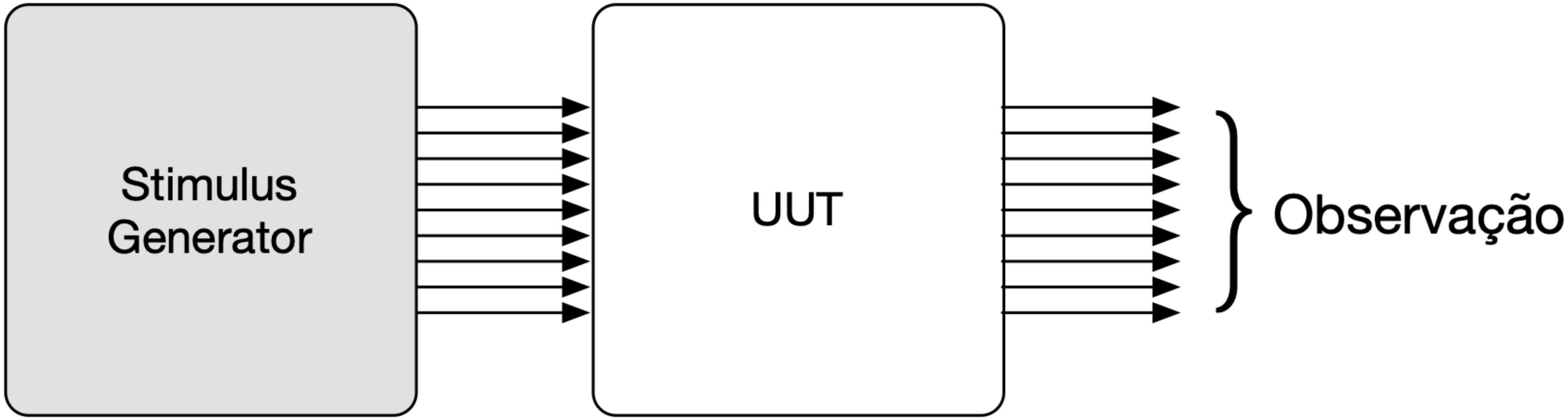
Name	Value	K	
X	0001	Signal	In
Y	0101	Signal	In
S	1	Signal	In
R	0101	Signal	Out
sa	0001	Signal	Internal
sx	0101	Signal	Internal

Name	Type (filtered)	State

	Msgs
/lucirc/X	0001
/lucirc/Y	0101
/lucirc/S	1
/lucirc/R	0101
/lucirc/sa	0001
/lucirc/sx	0101
Now	30 ns
Cursor 1	0.00 ns

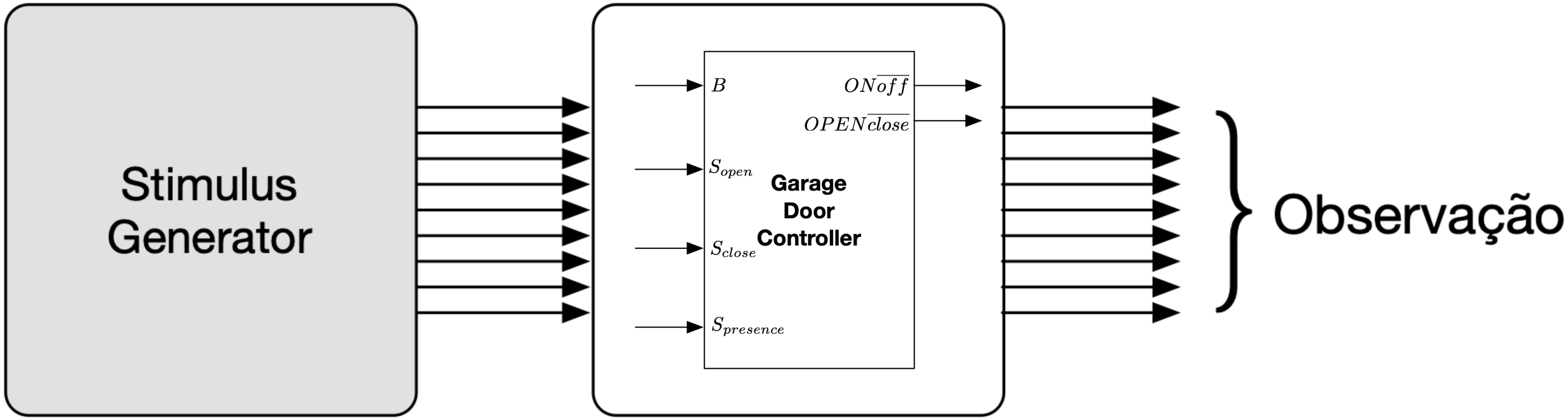
Teste Sistemático

- Unit Under Test

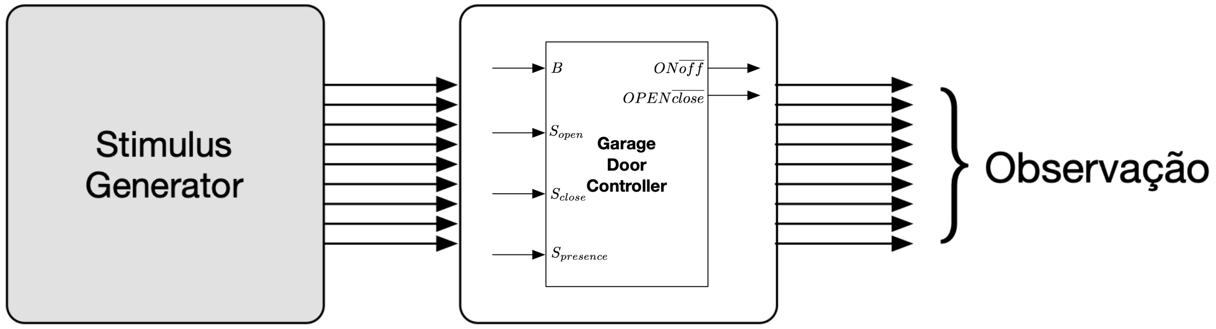


Teste Sistemático

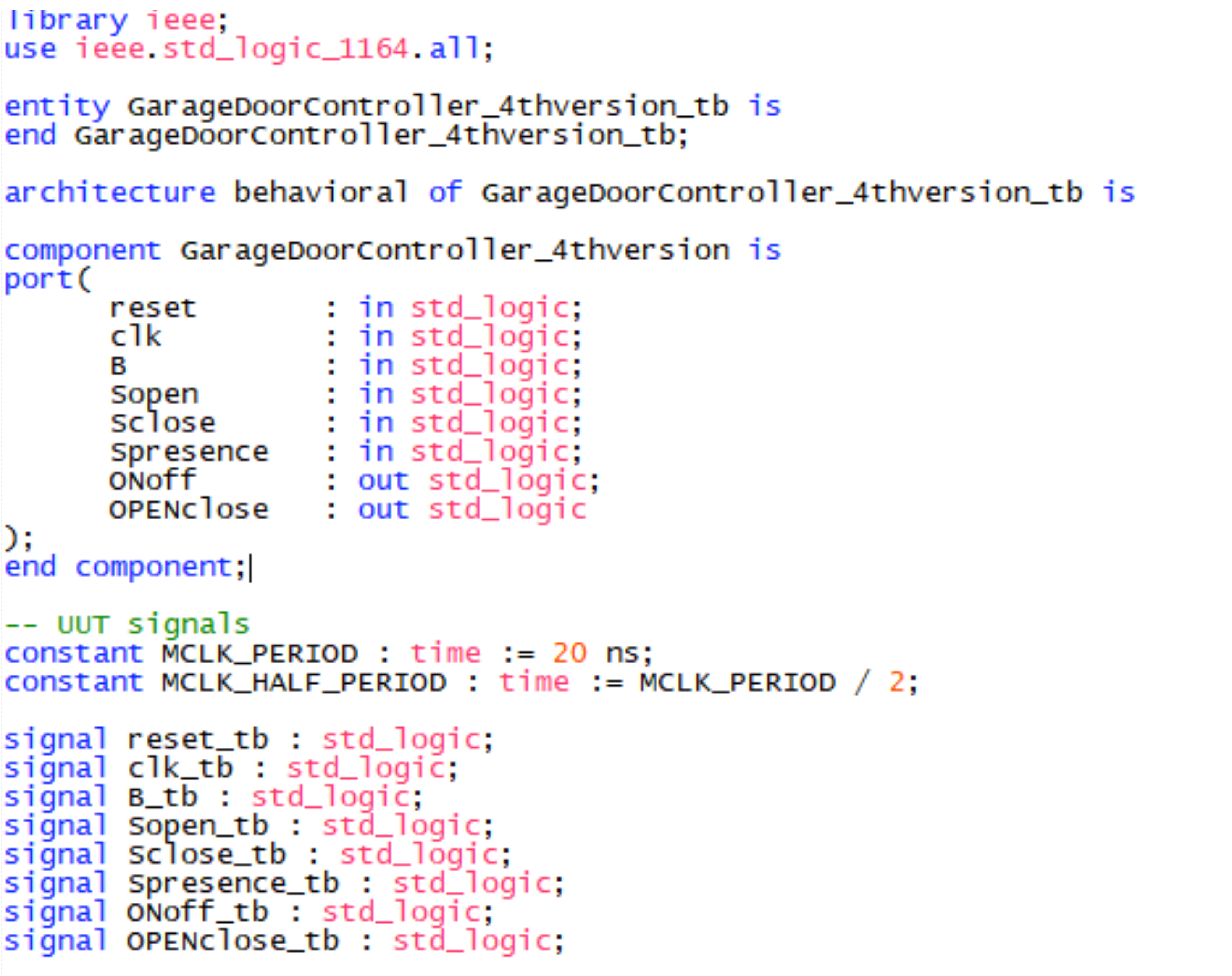
- Unit Under Test



Teste Sistemático



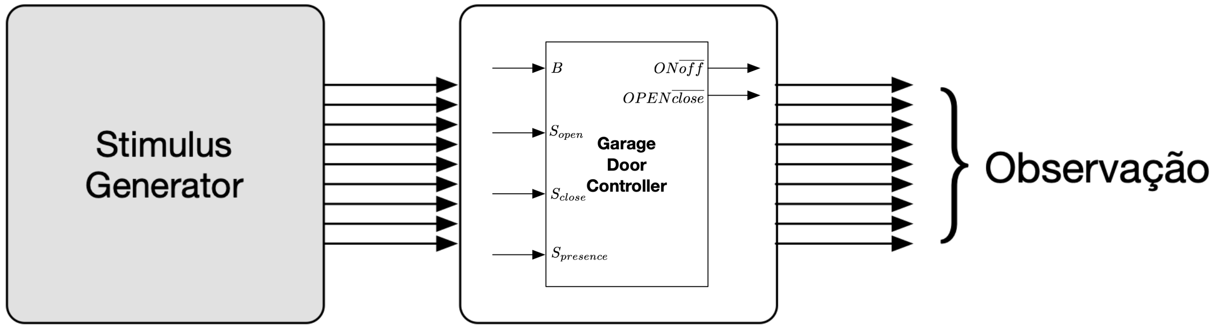
- Unit Under Test

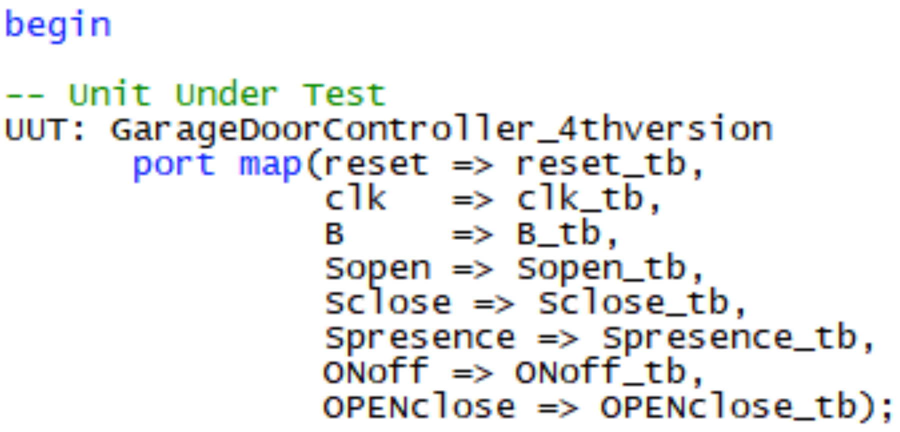


```
library ieee;
use ieee.std_logic_1164.all;

entity GarageDoorController_4thversion_tb is
end GarageDoorController_4thversion_tb;

architecture behavioral of GarageDoorController_4thversion_tb is

component GarageDoorController_4thversion is
port(
      reset       : in std_logic;
      clk         : in std_logic;
      B           : in std_logic;
      Sopen       : in std_logic;
      Sclose      : in std_logic;
      Spresence   : in std_logic;
      ONoff       : out std_logic;
      OPENclose   : out std_logic
);
end component;|

-- UUT signals
constant MCLK_PERIOD : time := 20 ns;
constant MCLK_HALF_PERIOD : time := MCLK_PERIOD / 2;

signal reset_tb : std_logic;
signal clk_tb : std_logic;
signal B_tb : std_logic;
signal Sopen_tb : std_logic;
signal Sclose_tb : std_logic;
signal Spresence_tb : std_logic;
signal ONoff_tb : std_logic;
signal OPENclose_tb : std_logic;
```

Teste Sistemático

- Unit Under Test

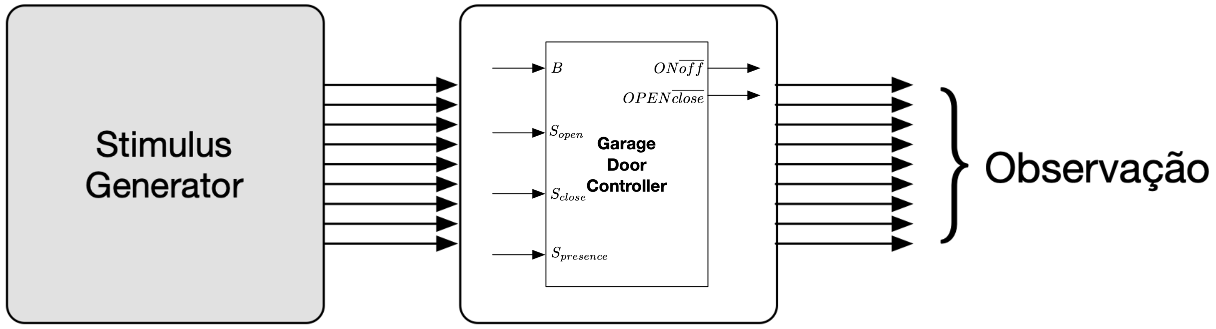


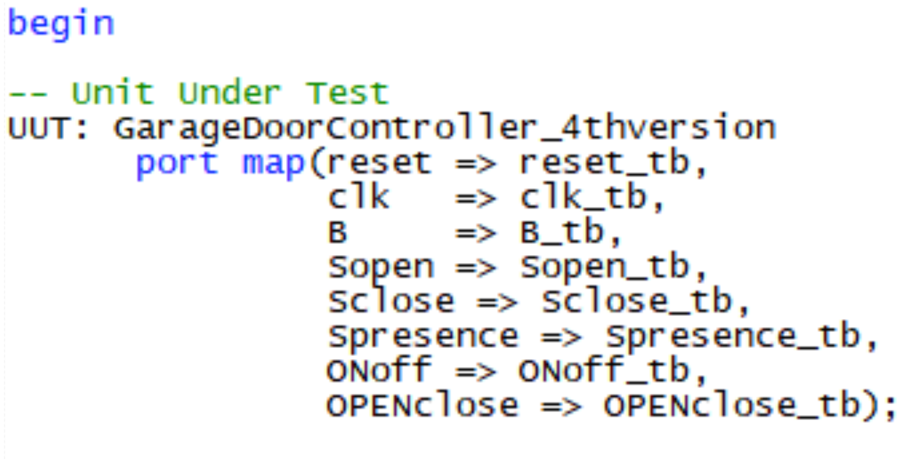


```
begin

-- Unit Under Test
UUT: GarageDoorController_4thversion
      port map(reset => reset_tb,
               clk   => clk_tb,
               B     => B_tb,
               Sopen => Sopen_tb,
               Sclose => Sclose_tb,
               Spresence => Spresence_tb,
               ONoff  => ONoff_tb,
               OPENclose => OPENclose_tb);
```

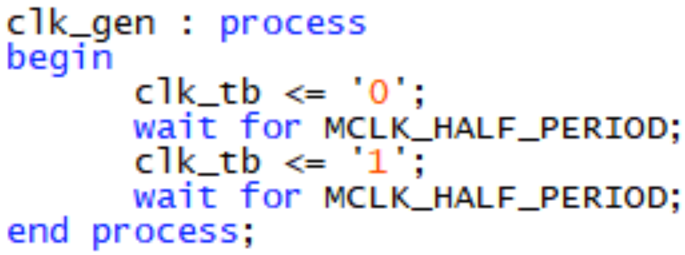
Teste Sistemático

- Unit Under Test





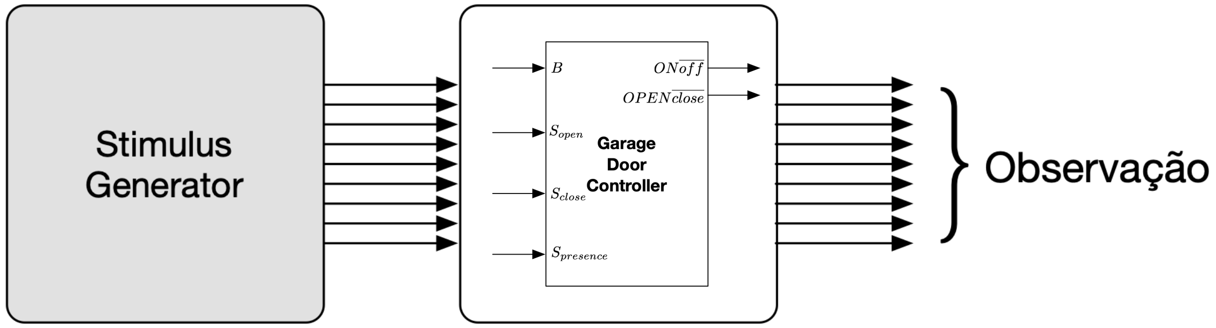
```vhdl
begin

-- Unit Under Test
UUT: GarageDoorController_4thversion
      port map(reset => reset_tb,
               clk   => clk_tb,
               B     => B_tb,
               Sopen => Sopen_tb,
               Sclose => Sclose_tb,
               Spresence => Spresence_tb,
               ONoff  => ONoff_tb,
               OPENclose => OPENclose_tb);
```

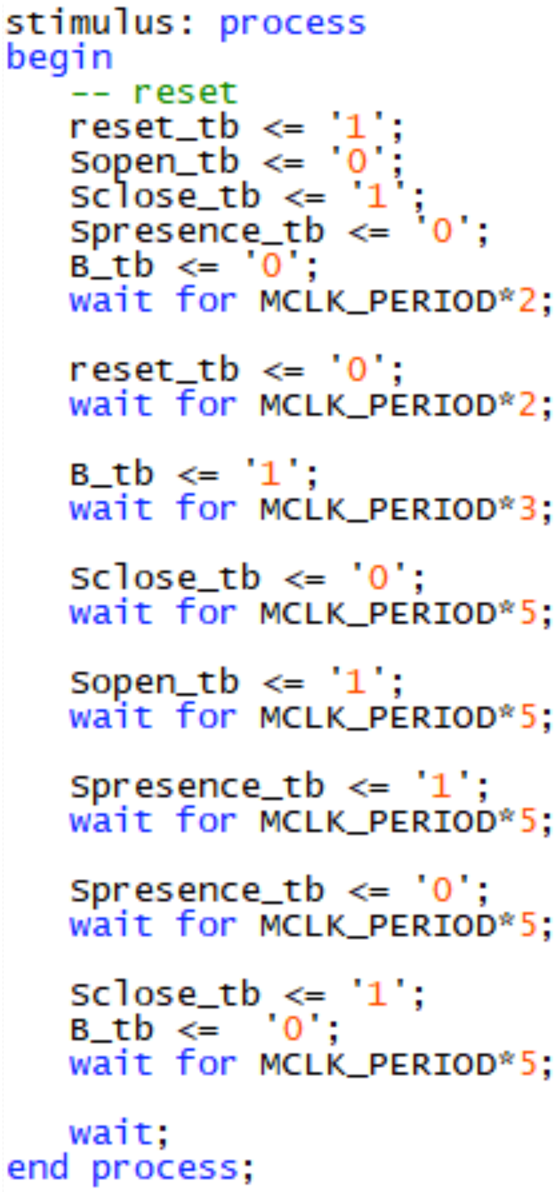


```vhdl
clk_gen : process
begin
      clk_tb <= '0';
      wait for MCLK_HALF_PERIOD;
      clk_tb <= '1';
      wait for MCLK_HALF_PERIOD;
end process;
```

Teste Sistemático

- Unit Under Test



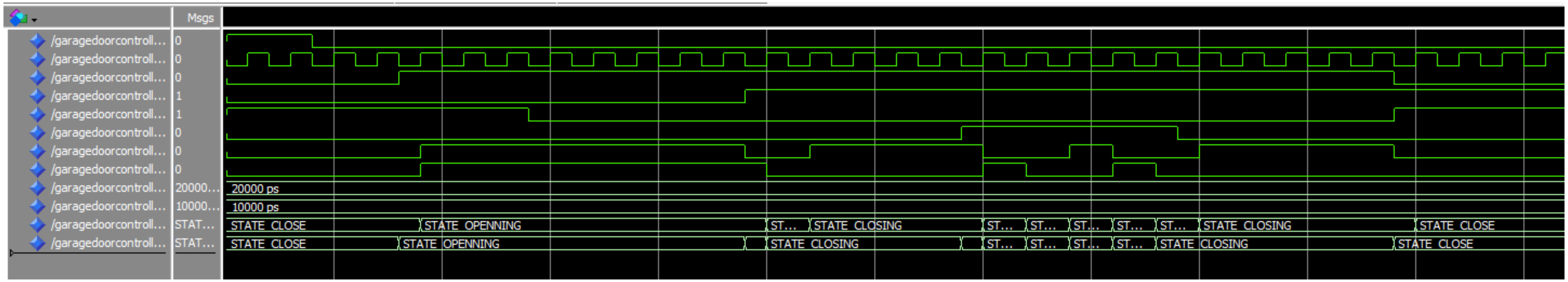


```
stimulus: process
begin
   -- reset
   reset_tb <= '1';
   Sopen_tb <= '0';
   Sclose_tb <= '1';
   Spresence_tb <= '0';
   B_tb <= '0';
   wait for MCLK_PERIOD*2;

   reset_tb <= '0';
   wait for MCLK_PERIOD*2;

   B_tb <= '1';
   wait for MCLK_PERIOD*3;

   Sclose_tb <= '0';
   wait for MCLK_PERIOD*5;

   Sopen_tb <= '1';
   wait for MCLK_PERIOD*5;

   Spresence_tb <= '1';
   wait for MCLK_PERIOD*5;

   Spresence_tb <= '0';
   wait for MCLK_PERIOD*5;

   Sclose_tb <= '1';
   B_tb <=  '0';
   wait for MCLK_PERIOD*5;

   wait;
end process;
```

Projeto: Controlo de Portão (4ª versão)

(*Garage Door Controller*)

- Projeto ASM-Chart


Msgs
/garagedoorcontroll... 0
/garagedoorcontroll... 0
/garagedoorcontroll... 0
/garagedoorcontroll... 1
/garagedoorcontroll... 1
/garagedoorcontroll... 0
/garagedoorcontroll... 0
/garagedoorcontroll... 0
/garagedoorcontroll... 20000...
/garagedoorcontroll... 10000...
/garagedoorcontroll... STAT...
/garagedoorcontroll... STAT...
20000 ps
10000 ps
STATE CLOSE
STATE OPENNING
ST...
STATE CLOSING
ST...
ST...
ST...
ST...
ST...
STATE CLOSING
STATE CLOSE
STATE CLOSE
STATE OPENNING
STATE CLOSING
ST...
ST...
ST...
ST...
STATE CLOSING
STATE CLOSE