# Casa Autônoma com FPGA

Hércules Ismael de Abreu Santos
Faculdade do Gama
Universidade de Brasília
Gama-DF, Brasil
ismael-456@hotmail.com
Github:https://github.com/ismaelg456g/Projeto_de_Circuitos_Reconfiguraveis_Hercules

Matheus Moreira da Silva Vieira
Faculdade do Gama
Universidade de Brasília
Gama-DF, Brasil
matheus.silvadf@gmail.com
Github: /matheusmsvieira

***Resumo* - Este projeto visa simular uma automação residencial com luzes externas e internas proporcionando praticidade e economia de energia.**
*Palavras-chave - Automação; VHDL; FPGA;*

## I. JUSTIFICATIVA

Considerando os crescentes avanços na automação residencial, cada vez mais quer-se obter mais facilidades para as tarefas do dia-a-dia. Desde tarefas como lavar roupas, lavar louças, ou mesmo fazer café, o que se vê é um avanço na maneira de se fazer estas tarefas, tornando-as cada vez mais fáceis. Com a automação de processos do cotidiano cada vez mais presentes, surge o termo "Casa Inteligente", para descrever a ideia de uma casa cada vez mais autônoma, gerando conforto, praticidade e segurança para o cotidiano.

Com isso em mente, deseja-se aplicar um sistema para controlar as luzes de uma casa com mais eficiência, de modo a gerar economia. A ideia consiste em controlar as luzes através de uma FPGA, e a partir de botões temporizar o acionamento das luzes de uma casa. O controle será feito de forma que se o usuário eventualmente esquecer de desligar as lâmpadas, como lâmpadas da área de serviço, ou jardim, que são facilmente esquecidas, o sistema então a partir de um limiar de tempo já pré-estabelecido irá desligar estas luzes, gerando economia desta forma. Além disso, o projeto contará com um sensor de gás para que reconheça a presença de gases nocivos como GLP (gás de cozinha) e monóxido de carbono, visto que muitos acidentes acontecem por pessoas esquecerem alguma boca do fogão aberta, mangueiras furadas ou, em estabelecimentos, câmaras de gás subterrâneas

## II. OBJETIVOS

Este projeto tem como objetivo a produção de um sistema de controle de luzes domésticas a partir de um sensor de presença e um temporizador e um detector de gases inflamáveis e/ou tóxicos.

## III. METODOLOGIA

Para executar isto, será necessário a especificação dos blocos lógicos necessários a essa aplicação. Vai ser necessário reconhecer o sensor como entrada além de uma chave simbolizando o interruptor, bloco divisor de clock e enviar um sinal para acender e apagar uma lâmpada. Depois será necessário a descrição do hardware em cada módulo, que será feita em VHDL. Então serão testados os módulos, para então uni-los, agregando ao protótipo, testá-lo, e adquirir como fim o protótipo final.

O funcionamento será da seguinte forma: Após o acionamento do interruptor um contador de 10 minutos irá iniciar a contagem. Após os 10 minutos o próprio contador vai acionar o sensor de presença que irá apagar a lâmpada se ele perceber uma inatividade de movimento por 5 minutos ininterruptos. Se o sensor reconhecer algum movimento dentro desses 5 minutos o contador irá zerar. Se o interruptor for acionado novamente a luz será apagada. O funcionamento do sensor de gás será da seguinte forma: se o sensor detectar a presença dos gases de cozinha ou monóxido de carbono o circuito fará com que as luzes pisquem de forma .Os testes serão feitos com tempos menores para facilitar a apresentação do projeto.

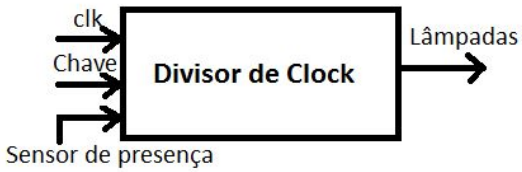


Fig. 1 - Diagrama de blocos da primeira parte do projeto

## IV. REVISÃO BIBLIOGRÁFICA

Muitos projetos e pesquisas são feitas nesta área, do qual podemos citar algumas aqui.

Usando FPGA, há um projeto da Sandip Foundation (SITRC), de Nashik na Índia [1], que utiliza FPGA como uma forma de automatizar casas. Seu sistema utiliza a FPGA como um controlador para automatizar vários aspectos de uma casa. Este projeto automatiza um ventilador, a partir de uma leitura de temperatura, as luzes da casa, a partir de um sensor de luminosidade, o LDR, implementa um sistema de alarme para vazamentos de gás, e controla a porta, trancando quando necessário. Todo este sistema se comunica por Wi-Fi com o celular do usuário, comunicando todas as ações executadas pelo sistema, e permitindo ao usuário ter controle do sistema.

Há também um projeto com controle de luzes de uma residência implementado em FPGA [2], na placa Basys 3, que aciona as luzes e temporiza seu desligamento, semelhante ao projeto que se quer implementar este.

Também pode se citar o uso de FPGA em automatização de sistemas de medição de umidade e temperatura controlados remotamente [3], em que são utilizados FPGA para automação.

## V. RESULTADOS

Para a primeira fase de testes do projeto, procurou-se obter a lógica em hardware, no VHDL, para controlar as luzes de uma residência como proposto no projeto.

Nesta fase porém, não se testará a lógica diretamente com o sensor a ser utilizado, e as lâmpadas. Para efeito de teste da lógica, utilizar-se-á os interruptores e leds disponíveis na própria placa em que se deseja desenvolver o projeto.

Assim, foram produzidos os blocos lógicos para a divisão do clock da placa, e o bloco que faz a contagem do tempo para desligar as luzes quando há inatividade prolongada na casa.

Neste teste, utilizamos uma divisão do clock para 1 Hz, de forma a obter uma contagem em segundos, assim ao se contar 10 segundos de inatividade, as luzes são desligadas automaticamente.

Também foi produzido um testbench, para se observar o comportamento da lógica na ferramenta do Vivado, de modo a obter uma visualização mais clara da lógica. Para o testbench reduziu-se a divisão do clock para um fator de 2, de modo que a frequência utilizada na simulação foi de 50 MHz, esta alteração deve ser feita para se conseguir simular na ferramenta, pois simular muitos segundos de comportamento é computacionalmente dispendioso para a ferramenta.

O próximo passo então, deve ser a conexão do sensor de movimento e das lâmpadas, através de um optoaclopador, que servirá para isolar a alta tensão das lâmpadas da baixa tensão da FPGA.

Assim, partindo para o próximo passo, foi feito um algoritmo em VHDL para fazer a interface do entre a FPGA e os sensores e a lâmpada a ser acionada. Foi adicionado a esse código a interface entre um sensor de gás, para detectar possíveis vazamentos de gases inflamáveis, acionando assim um led de alarme para avisar do perigo iminente de tal vazamento.

O acionamento da lâmpada foi feito através de uma porta GPIO (Porta digital de uso geral) da placa, que se liga à lâmpada por meio de um optoacoplador, ligado à um relé, que aciona a lâmpada, isso porque a FPGA não opera em corrente alternada, como uma lâmpada comum de uso doméstico, por isso é necessário isolar o circuito de controle, que inclui a FPGA, do circuito de força, que é o circuito para se acionar a lâmpada.

Os sensores também foram interfaceados com a basys 3, através de portas GPIO.

## REFERÊNCIAS

[1] SHARMA, Sanjeev; DEOKAR, Revati. FPGA Based Cost Effective Smart Home Systems. In: **2018 International Conference On Advances in Communication and Computing Technology (ICACCT)**. IEEE, 2018. p. 397-402.

[2] ALEX, “FPGA Automated House Lights”, Disponível em:

<https://www.instructables.com/id/FPGA-Automated-House-Lights/> Acesso em 10 abr. 2019
[3] EL-MEDANY, Wael M. FPGA implementation for humidity and temperature remote sensing system. In: **2008 IEEE 14th International Mixed-Signals, Sensors, and Systems Test Workshop**. IEEE, 2008. p. 1-4.

## APÊNDICE

**Código em VHDL do primeiro teste:**

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity top_level is
    Port ( clk : in STD_LOGIC;
           sw : in STD_LOGIC_VECTOR (3 downto
0);
           luzes : out STD_LOGIC_VECTOR (7
downto 0);
           sensor : in STD_LOGIC);
end top_level;

architecture Behavioral of top_level is
    signal cont, cont_inat: integer:=0;
    signal clk_1Hz: std_logic:='0';
    signal desliga_luz: std_logic:='1';
begin
    div_clk:process(clk)
    begin
        if rising_edge(clk) then
            if(cont>=50000000)then
                clk_1Hz<=not clk_1Hz;
                cont<=0;
            else
                cont<=cont+1;
            end if;
        end if;
    end process;

    cont_inatividade:process(clk_1Hz)
    begin
        if(rising_edge(clk_1Hz))then
            if sensor='1'then
                if cont_inat>=10 then
                    desliga_luz<='0';
                else
                    desliga_luz<='1';
                    cont_inat<=cont_inat+1;
                end if;
            else
                desliga_luz<='1';
                cont_inat<=0;
            end if;
        end if;
    end process;

    luzes(0)<= sw(0) and desliga_luz;
    luzes(1)<= sw(0) and desliga_luz;
    luzes(2)<= sw(1) and desliga_luz;
    luzes(3)<= sw(1) and desliga_luz;
    luzes(4)<= sw(2) and desliga_luz;
    Luz es(5)<= sw(2) and desliga_luz;
    luzes(6)<= sw(3) and desliga_luz;
    luzes(7)<= sw(3) and desliga_luz;

end Behavioral;
```

**Código em VHDL para o testbench do primeiro teste:**

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity tb_top_level is
--  Port ( );
end tb_top_level;

architecture Behavioral of tb_top_level is

component top_level is
    Port ( clk : in STD_LOGIC;
           sw : in STD_LOGIC_VECTOR (3 downto
0);
           luzes : out STD_LOGIC_VECTOR (7
downto 0);
           sensor : in STD_LOGIC);
```

```
end component;

signal clk: std_logic;
signal sw: std_logic_vector(3 downto 0);
signal luzes: std_logic_vector(7 downto 0);
signal sensor: std_logic;

begin

uut: top_level port map(clk,sw,luzes,sensor);

gen_clk:process
begin
   clk<='1';
   wait for 5 ns;
   clk<='0';
   wait for 5 ns;
end process;

stimulus:process
begin
   sw<="1111";sensor<='1';
   wait for 300 ns;
   sensor<='0';
   wait;
end process;

end Behavioral;
```

**Código em VHDL do segundo teste para utilizar na fpga com sensor de presença e sensor de gás:**

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity top_level is
   Port ( clk : in STD_LOGIC;
          sw : in STD_LOGIC;
          luzes : out STD_LOGIC;
          led_alarme : out STD_LOGIC;
          sensor_gas : in STD_LOGIC;
          sensor_presenca : in STD_LOGIC);
end top_level;

architecture Behavioral of top_level is
   signal cont1Hz, cont4Hz, cont_inat: integer:=0;
   signal clk_1Hz: std_logic:='0';
   signal clk_4Hz: std_logic:='0';
   signal desliga_luz: std_logic:='1';
   signal s_led_alarme: std_logic:='0';
begin
   div_clk:process(clk)
   begin
      if rising_edge(clk) then
         if(cont4Hz>=12500000)then
            clk_4Hz<=not clk_4Hz;
            cont4Hz<=0;
         else
            cont4Hz<=cont4Hz+1;
         end if;

         if(cont1Hz>=50000000)then
            clk_1Hz<=not clk_1Hz;
            cont1Hz<=0;
         else
            cont1Hz<=cont1Hz+1;
         end if;
      end if;
   end process;

   cont_inatividade:process(clk_1Hz)
   begin
      if(rising_edge(clk_1Hz))then
         if sensor_presenca='0'then
            if cont_inat>=10 then
               desliga_luz<='0';
            else
               desliga_luz<='1';
               cont_inat<=cont_inat+1;
            end if;
         else
            desliga_luz<='1';
            cont_inat<=0;
         end if;
```

```
    end if;
  end process;

  proc_alarm: process(clk_4Hz)
  begin
    if rising_edge(clk_4Hz) then
      if sensor_gas='1' then
        s_led_alarme<= not s_led_alarme;
      else
        s_led_alarme<= '0';
      end if;
    end if;
  end process;



  luzes<= sw and desliga_luz;
  led_alarme<= s_led_alarme;

end Behavioral;
```

**Código do xdc do segundo teste para utilizar na fpga com sensor de presença e sensor de gás:**

```
## Clock signal
set_property PACKAGE_PIN W5 [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
set_property PACKAGE_PIN V17 [get_ports
{sw}]
        set_property IOSTANDARD LVCMOS33
[get_ports {sw}]

## LEDs
set_property PACKAGE_PIN U16 [get_ports
{led_alarme}]
        set_property IOSTANDARD LVCMOS33
[get_ports {led_alarme}]

##Pmod Header JA
#Sch name = JA1
set_property PACKAGE_PIN J1 [get_ports
{luzes}]
        set_property IOSTANDARD LVCMOS33
[get_ports {luzes}]
#Sch name = JA2
set_property PACKAGE_PIN L2 [get_ports
{sensor_gas}]
        set_property IOSTANDARD LVCMOS33
[get_ports {sensor_gas}]
#Sch name = JA3
set_property PACKAGE_PIN J2 [get_ports
{sensor_presenca}]
        set_property IOSTANDARD LVCMOS33
[get_ports {sensor_presenca}]
```