

Relatório de Sistemas de Aquisição de Dados
2006/2007

Engenharia Física Tecnológica

**AMOSTRAGEM DE SINAIS ANALÓGICOS
VARIÁVEIS POR UM CIRCUITO ADC COM
INTERFACE DE COMUNICAÇÃO SÉRIE**

Laboratório III

Trabalho realizado por:

Alexandra Gouveia, n°53760

João Mendes Lopes, n°53788

André Cunha, n°53757

Grupo 3; 4ªfeira 15:00-19:00h

Lisboa, 16 de Junho de 2007

O objectivo primordial deste trabalho consiste na aquisição de uma série de amostras periódicas de um sinal variável, através do circuito ADC de interface série síncrona. Para tal, pretende-se utilizar o protocolo SPI para a comunicação de uma FPGA SPARTAN 3 com um dispositivo ADC, desenvolvido no trabalho de laboratório anterior, mas adaptado para adquirir amostras individuais periódicas de um sinal analógico variável sinusoidal, armazenando diversos valores durante um determinado tempo e permitindo a visualização dos diversos valores no mostrador LCD/LED, através dos *switches* da placa. Os sinais adquiridos devem ser comparados com o sinal analógico e ainda efectuadas as transformadas FFT dos mesmos.

Para aquisição dos valores através do ADC, é necessário um protocolo de comunicação entre o dispositivo de conversão analógica digital, e o circuito que rege o seu funcionamento. À semelhança do laboratório anterior, o ADC usado (MCP3202) rege-se pelo protocolo SPI. O trabalho que se pretende desenvolver no laboratório III é, na verdade, um seguimento lógico do que foi desenvolvido anteriormente, pois a diferença entre os dois trabalhos consiste na adaptação do projecto para que este possa adquirir diversos valores de um sinal variável, ao invés de um único de sinal constante e os possa guardar numa memória, acedida posteriormente através da manipulação de *switches* na placa. Como tal, e como o funcionamento do protocolo SPI e do ADC MCP3202 já foram explicados no relatório anterior (ver bibliografia), não haverá um alongamento sobre esta matéria, de modo a evitar repetição de informação e redundância. Pretende-se focar as diferenças entre os dois projectos e destacar a forma como o primeiro projecto foi alterado para a criação deste.

1ª sessão de laboratório

Material utilizado:

- Ambiente de programação XILINX ISE “MPLAB” e compilador “VHDL”.

A primeira sessão de laboratório consistiu na alteração do trabalho de laboratório nº 2, de acordo com os requisitos indicados anteriormente. Para tal, criou-se um novo sinal externo, um vector de cinco elementos do tipo entrada chamado *switch*, para que as alterações feitas nos *switches* da placa pudessem ser reconhecidas no interior do programa. Apesar de existirem oito *switches* na placa SPARTAN 3, para ler nestes dispositivos um valor até 32 bastava utilizar os primeiros cinco e daí a dimensão do vector. Para o programa, e de acordo com o funcionamento da placa descrito no manual da mesma, um *switch* ligado vale 1 e um *switch* desligado vale 0. Para além disso, criou-se o sinal interno *indice*, um natural que pudesse ser incrementado à medida que era feita cada uma das aquisições, não só como um sinal de controlo para ter um critério de paragem tal que quando *indice* atingisse o valor 32 parassem as aquisições, como também para servir de índice para a transferência de cada uma das aquisições para posições de memória específicas. Foi ainda criado um sinal interno *vector* de doze elementos, cuja função é descrita adiante e uma memória, um *array* de 32 vectores cada um com doze elementos. Posteriormente o sinal *indice* foi inicializado a 0 no *reset* existente no processo que implementa o protocolo SPI. De seguida tentou-se que o bloco de código que implementa este protocolo corresse 32 vezes mediante um comando de *if* que postula que enquanto o índice for menor que 32 o bloco de código deve ser executado. No final de cada execução do protocolo SPI, no instante onde no 2º trabalho de laboratório se voltava a por *cs_FPGA* a *high* acrescentaram-se algumas linhas de código, nomeadamente: a aquisição é passada para a posição de memória correspondente ao valor do índice, o contador de 50kHz é reinicializado, para que na segunda aquisição os passos dependentes deste contador se voltem a executar, o índice é incrementado, o sinal *state* é reinicializado a 0 para que a nova aquisição se possa desencadear e a palavra de controlo é reinicializada a 1101, caso contrário o programa ‘esquece’ o seu valor. Para além disto, *cs_FPGA* continua a ser posto a *high* neste instante.

Assim, mediante este raciocínio, esperava-se que o protocolo fosse executado 32 vezes e as aquisições sucessivas guardadas em posições de memória sucessivas. De seguida, foi necessário fazer com que o programa lesse o valor dos *switches* da placa e consoante o seu valor mostrasse a aquisição correspondente no *display* de sete segmentos. Para tal, acrescentou-se um *elsif* ao programa, para que quando o índice valesse 32 fosse executada esta parte do programa e não se sáisse mais deste ‘estado’. Nele acrescentou-se o comando

```
vector<=tmp_ram(conv_integer(switch));
```

que lê o valor dos *switches* na placa, converte o seu valor para um inteiro e envia a posição de memória correspondente ao valor dos *switches* para o sinal interno *vector*. Inicialmente, este comando consistia num *case*, que para cada valor dos *switches* enviava o conteúdo da posição respectiva de memória para *vector*, mas o comando indicado acima cumpre o mesmo objectivo, poupando-se muitas linhas de código.

Projecto e implementação das alterações ao 2º trabalho de laboratório

De seguida bastou alterar o programa, para que em vez de se enviarem os troços do vector *AQUISICAO* para o sinal *curr*, que depois modifica o valor da variável *seg* mediante o descodificador, processo descrito no anterior trabalho de laboratório, se enviarem os troços correspondentes do sinal *vector*. Quando o programa acaba de enviar os sinais de endereçamento para os *displays*, o valor de *switch* volta a ser verificado e o código repete-se indefinidamente, o que permite que cada alteração dos *switches* da placa seja percebida pelo programa, que age em conformidade. O módulo descodificador permaneceu inalterado em relação ao que havia sido implementado no 2º trabalho de laboratório.

Assim, o código final deste projecto, construído da forma descrita era o seguinte:

```
--importação de bibliotecas
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

--declaração de variáveis externas
entity SPI is
    Port ( din_FPGA : in STD_LOGIC;
          dout_FPGA : out STD_LOGIC;
          reset : in STD_LOGIC;
          cs_FPGA : out STD_LOGIC;
          clock50Mhz : in STD_LOGIC;
          clockSPI : out STD_LOGIC;
          switch : in std_logic_vector(4 downto 0) ;
          seg : out std_logic_vector(6 downto 0) ;
          digit : out std_logic_vector(3 downto 0));
end SPI;

architecture Behavioral of SPI is

--importação do modulo descodificador
COMPONENT dec_seg is
    Port ( char : in STD_LOGIC_VECTOR (3 downto 0);
          seg : out STD_LOGIC_VECTOR (6 downto 0));
end COMPONENT;

--definição de variáveis internas
signal PALAVRA: STD_LOGIC_VECTOR (3 downto 0);
signal AQUISICAO: STD_LOGIC_VECTOR (12 downto 0);
signal clock50kHz : STD_LOGIC;
signal count50kHz : STD_LOGIC_VECTOR (24 downto 0);
signal count: STD_LOGIC_VECTOR (24 downto 0);
signal state : STD_LOGIC_VECTOR (1 downto 0);
signal dout_tmp : STD_LOGIC;
signal cs_tmp : STD_LOGIC;
signal curr : STD_LOGIC_VECTOR (3 downto 0);
signal vector : std_logic_vector(11 downto 0);
signal indice : natural;
```

Projecto e implementação das alterações ao 2º trabalho de laboratório

```
--variável interna tipo state
type STATE_TYPE is (S1, S2, S3, S4, S5, S6, S7, S8);
signal stateDisp, nextState: STATE_TYPE;

--declaração de um array de 31 vectores com 12 elementos
type ram_type is array (0 to 31) of
    std_logic_vector(11 downto 0);
signal tmp_ram: ram_type;

begin

--identificação de clockSPI com clock50kHz
clockSPI <= clock50kHz;

--importação do módulo do decodificador
decoder:    dec_seg PORT MAP(
            char=>curr,
            seg=>seg
            );

--implementação de um relógio de 50kHz
main:      process (clock50Mhz, reset)
begin

    if reset='1' then
        count <= (others =>'0');
        clock50kHz <= '1';

    elsif rising_edge(clock50Mhz) then

        count <= count + 1;

        if count = 500 then
            clock50kHz <= '0';
        end if;

        if count = 1000 then
            clock50kHz <= '1';
            count <= (others =>'0');
        end if;

    end if;

end process;

--implementação do protocolo SPI

--actualização de dout_FPGA e cs_FPGA em transições de
--flanco negativo do relógio SPI

process (clock50kHz)
begin

    if falling_edge(clock50kHz) then
        dout_FPGA <= dout_tmp;
```

```

        cs_FPGA <= cs_tmp;
    end if;
end process;

process (reset, clock50kHz)
begin

--inicializações
    if reset='1' then
        state <= "00";
        PALAVRA <= "1101";
        count50kHz <= (others =>'0');
        cs_tmp <= '1';
        digit <= (others => '1') ;
        curr <= (others => '0') ;
        nextState <= S1;
        indice <= 0;

--para cada flanco positivo de relógio

        elsif rising_edge(clock50kHz) then

--enquanto o índice for menor que 32 são feitas 32
--aquisições pelo ADC

            if indice < 32 then

--contador incrementa
                count50kHz <= count50kHz + 1;

--no estado 0 iniciam-se as comunicações através da
--colocação de cs_tmp a low, envia-se o start bit (primeiro bit da
--palavra de controlo do ADC) e muda-se de estado

                if state= "00" then

                    if count50kHz = 1 then
                        cs_tmp <= '0';
                        dout_tmp <= PALAVRA(3);
                        PALAVRA(3 downto 1) <= PALAVRA (2 downto 0);
                        State <= "01";
                    end if;

                end if;

--no estado 1 enviam-se os restantes bits de controlo e
--muda-se de estado

                if state="01" then
                    dout_tmp <= PALAVRA(3);
                    PALAVRA(3 downto 1) <= PALAVRA (2 downto 0);

                    if count50kHz = 4 then
                        state <= "10";
                    end if;

                end if;

            end if;

```

Projecto e implementação das alterações ao 2º trabalho de laboratório

```
--no estado 2 recebe-se a palavra do ADC; de seguida passa-se
--cada aquisição para uma posição de memória definida pelo índice
--(1º aquisição para a 1º posição de memória, etc.) põe-se cs a
--high para que cessem as comunicações, incrementa-se o índice,
--reinicia-se o vector PALAVRA com a palavra de comando
--bem como o contador de 50MHz e volta-se ao estado inicial
```

```
if state = "10" then
  AQUISICAO(0) <= din_FPGA ;
  QUISICAO(12 downto 1) <= AQUISICAO(11 downto 0);

  if count50kHz = 18 then
    stateDisp<= S1;
    tmp_ram (indice) <= AQUISICAO(11 downto 0);
    count50kHz <= (others =>'0');
    indice<=indice+1;
    state <= "00";
    cs_tmp <='1';
    PALAVRA<="1101";
  end if;
end if;
```

```
--quando foram feitas 32 aquisições cessam definitivamente
--as comunicações com o ADC e é mostrada no display uma das 32
--aquisições consoante a posição dos switches da placa.
--Permaneçe-se neste estado indefinidamente, passando-se
--constantemente a aquisição requerida para os displays
```

```
elsif indice = 32 then
```

```
--é passada para um vector temporário a aquisição que se pretende
--visualizar
```

```
vector<=tmp_ram(conv_integer(switch));
```

```
--em cada flanco positivo o sinal stateDisp é
--actualizado para nextState
```

```
stateDisp <=nextState;
```

```
--no primeiro estado, é escrito um A no display AN3
```

```
if stateDisp = S1 then
  curr <= "1010";
  digit <= "0111";
  nextState <= S2;
```

```
--no segundo estado, o display AN3 é apagado
```

```
elsif stateDisp = S2 then
  digit <= "1111";
  nextState <= S3;
```

Projecto e implementação das alterações ao 2º trabalho de laboratório

```
--no terceiro estado, são escritos os bits das posições 11
--a 8 da aquisição no display AN2
    elsif stateDisp = S3 then
        curr <= vector(11 downto 8);
        digit <= "1011";
        nextState <= S4;

--no quarto estado, o display AN2 é apagado
    elsif stateDisp = S4 then
        digit <= "1111";
        nextState <= S5;

--no quinto estado, são escritos os bits na posição 7 a 4
--da aquisição no display AN1
    elsif stateDisp = S5 then
        curr <= vector(7 downto 4);
        digit <= "1101";
        nextState <= S6;

--no sexto estado, o display AN1 é apagado
    elsif stateDisp = S6 then
        digit <= "1111";
        nextState <= S7;

--no sétimo estado, são escritos os quatro bits menos
--significativos da aquisição no display AN0
    elsif stateDisp = S7 then
        curr <= vector(3 downto 0);
        digit <= "1110";
        nextState <= S8;

--no oitavo estado, o display AN0 é apagado
    elsif stateDisp = S8 then
        digit <= "1111";
        nextState <= S1;

    end if;

--o programa mantém-se nesta rotina indefinidamente,
--enviando continuamente sinais de endereçamento para os
--displays

end if;

end process;

end Behavioral;
```

Projecto e implementação das alterações ao 2º trabalho de laboratório

Apesar do modulo descodificador não se ter alterado, permanecendo igual ao implementado para o 2º trabalho de laboratório, é aqui incluído para que o código fique completo:

```
--importação de bibliotecas
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

entity dec_seg is
    Port ( char : in  STD_LOGIC_VECTOR (3 downto 0);
          seg  : out STD_LOGIC_VECTOR (6 downto 0));
end dec_seg;

architecture Behavioral of dec_seg is

begin

--codificador para a programação dos sete segmentos
    process (char)
        begin
--a cada valor de curr corresponde uma representação em hexadecimal
            case char is
                when "0000" => seg <= "0000001" ;--0
                when "0001" => seg <= "1001111" ;--1
                when "0010" => seg <= "0010010" ;--2
                when "0011" => seg <= "0000110" ;--3
                when "0100" => seg <= "1001100" ;--4
                when "0101" => seg <= "0100100" ;--5
                when "0110" => seg <= "0100000" ;--6
                when "0111" => seg <= "0001111" ;--7
                when "1000" => seg <= "0000000" ;--8
                when "1001" => seg <= "0000100" ;--9
                when "1010" => seg <= "0001000" ;--A
                when "1011" => seg <= "1100000" ;--b
                when "1100" => seg <= "0110001" ;--C
                when "1101" => seg <= "1000010" ;--d
                when "1110" => seg <= "0110000" ;--E
                when "1111" => seg <= "0111000" ;--F
                when others => seg <= "1111110" ;
            end case ;
        end process;

end Behavioral;
```

Projecto e implementação das alterações ao 2º trabalho de laboratório

Depois de implementado o projecto, este foi testado, recorrendo ao habitual ficheiro de teste. Como no 2º trabalho de laboratório já se tinha verificado o bom funcionamento da implementação do protocolo SPI, bem como do envio dos sinais necessários aos *displays* de sete segmentos para que o valor da aquisição possa ser aí visualizado, pretendia-se apenas confirmar que estes elementos do programa continuavam a funcionar bem e que para cada uma das 32 aquisições agora implementadas o valor da aquisição ia para a posição de memória pretendida. Também se quis verificar o bom funcionamento dos *switches*. Como na simulação a palavra *AQUISICAO* ainda não se encontra definida, para fazer o teste, definiu-se um valor para este vector “0100010100011” e comentou-se o código que fazia a transferência de *din_FPGA* para o mesmo. Para esta parte do projecto verificou-se o bom funcionamento do circuito simulado, encontrando-se o resultado da simulação na **Figura 1**

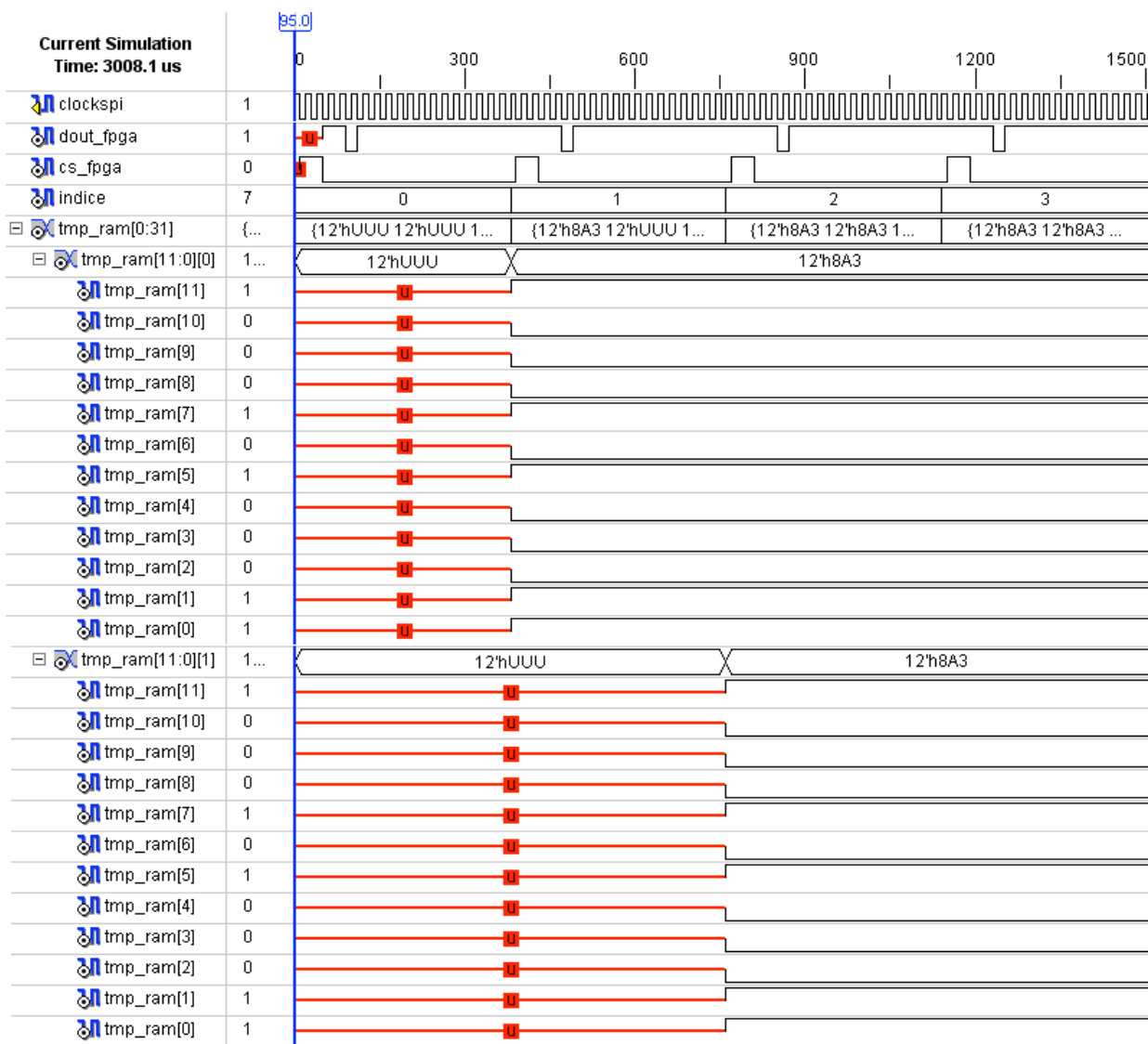


Figura 1 – Simulação do funcionamento da memória

Pode-se verificar o bom funcionamento do programa, já que o vector definido para a *AQUISICAO* é colocado em cada ciclo do programa numa posição diferente de memória. Chama-se a atenção para o facto de ser precisamente no momento em que a aquisição é colocada na memória que o índice é incrementado e daí que só no final da etapa em que o índice é 0 é que a aquisição é colocada na posição 0 de memória e assim sucessivamente. Pode-se ainda observar o correcto funcionamento de *cs_FPGA* e *dout_FPGA* que variam da forma pretendida e para flancos negativos de relógio. Embora não se consiga ver a continuação da simulação, verificou-se que este padrão se repetia até que o índice atingia o valor 32. Nesta altura a última posição de memória é preenchida e passa-se à segunda fase do programa, de endereçamento dos *displays*, verificando-se que o padrão de endereçamento como referido no 2º trabalho de laboratório se repetia indefinidamente, como se pretendia. Pode-se verificar que a palavra enviada para definir o valor apresentado nos *displays* de sete segmentos correspondia alternadamente A, 3, A e 8, sendo o primeiro A definido directamente no programa como foi indicado no 2º trabalho de laboratório e os restantes valores os que se esperavam dado o valor que se definiu para a aquisição. É de lembrar que o bit mais significativo deste vector é ignorado, por corresponder ao *null bit* sem valor informativo enviado pelo ADC antes de enviar a aquisição propriamente dita.

Uma vez verificado o bom funcionamento da memória foi necessário verificar se o sistema de visualização da aquisição através da manipulação dos *switches* estava a funcionar bem. Como os *switches* só podiam ser realmente alterados na placa, mudou-se a declaração do vector *switch* para ser uma variável *inout* e definiu-se antes da passagem da aquisição para o vector temporário um valor para a variável *switch* e outro para a primeira posição de memória.

```
switch<="00000";  
tmp_ram (0) <= "010111101011";
```

em que os troços de cada quatro bits definidos para a primeira posição de memória correspondem a 5, E e b em hexadecimal. Com este procedimento verificou-se o bom funcionamento do circuito simulado para esta parte do projecto, encontrando-se o resultado da simulação na **Figura 2**.

Projecto e implementação das alterações ao 2º trabalho de laboratório

Uma vez verificado o bom funcionamento do circuito simulado, voltaram-se a fazer as alterações necessárias para que o programa ficasse no seu estado inicial.

É importante salientar que uma vez tendo o projecto referente ao 2º trabalho de laboratório a funcionar, foi relativamente simples executar as alterações descritas e implementar correctamente o projecto em questão, não havendo dificuldades a apontar.

Aquisição de um sinal variável com um ADC com interface SPI MPC3202

2ª sessão de laboratório

Material utilizado:

- Ambiente de programação XILINX ISE “MPLAB” e compilador “VHDL”;
- Kit SPARTAN 3;
- Multímetro;
- Osciloscópio;
- Ligações unifilares;
- Breadboard*;
- Fonte de alimentação DC e gerador de sinais;
- Dispositivo ADC de 12 bits do tipo aproximações sucessivas com interface SPI MPC3202.

Uma vez verificado o bom funcionamento do código implementado, pretendiam-se obter 32 amostras individuais de um sinal analógico sinusoidal para cada pressão manual no botão de *reset* da placa e visualizar essas aquisições no LCD da placa SPARTAN 3 consoante o valor dos *switches* manipulados pelo utilizador na mesma placa. Esta dispõe de oito *switches*, mas para representar até 32 valores bastava manipular os primeiros cinco, sendo os restantes ignorados pelo programa.

Assim, programou-se o ficheiro .ucf, editado correctamente para o 2º trabalho de laboratório, tendo sido apenas necessário acrescentar os portos relativos aos cinco *switches* a utilizar e modificar uma das saídas da placa, porque se verificou que o porto A3, correspondente à saída da FPGA *clockSPI* não estava a funcionar. O ficheiro depois de configurado era o seguinte:

```
NET "clock50Mhz" LOC = "T9";
NET "reset"      LOC = "L14";
NET "digit<0>"  LOC = "D14";
NET "digit<1>"  LOC = "G14";
NET "digit<2>"  LOC = "F14";
NET "digit<3>"  LOC = "E13";
NET "seg<0>"    LOC = "N16";
NET "seg<1>"    LOC = "F13";
NET "seg<2>"    LOC = "R16";
NET "seg<3>"    LOC = "P15";
```

Aquisição de um sinal variável com um ADC com interface SPI MPC3202

```
NET "seg<4>"      LOC = "N15";
NET "seg<5>"      LOC = "G13";
NET "seg<6>"      LOC = "E14";
NET "din_FPGA"    LOC = "C5";
NET "dout_FPGA"   LOC = "C7";
NET "cs_FPGA"     LOC = "C9";
NET "clockSPI"    LOC = "D10";
NET "switch<0>"  LOC = "F12";
NET "switch<1>"  LOC = "G12";
NET "switch<2>"  LOC = "H14";
NET "switch<3>"  LOC = "H13";
NET "switch<4>"  LOC = "J14";
```

Posteriormente colocou-se o ADC na *breadboard* ligando-o à placa SPARTAN 3 com as linhas de comunicação necessárias, um procedimento já descrito no 2º trabalho de laboratório. Antes de se ligar o gerador de sinais ao ADC teve-se o cuidado de evitar que o sinal gerado tivesse uma amplitude negativa ou superior a 3.3V. Para tal, antes de ligar qualquer sinal à entrada do ADC, ligou-se a massa do osciloscópio a um sinal constante a -1V medido com cuidado com o auxílio de um multímetro e regulou-se a amplitude do sinal sinusoidal de forma a que onda observada no osciloscópio não tivesse troços negativos, obtendo-se desta forma um sinal com amplitude de 0 a 2V.

Uma vez obtido um sinal sinusoidal com as características necessárias, pode-se ligar a massa da placa ao sinal de -1V da base de experimentação e o sinal sinusoidal à entrada do ADC. Para se verificar se a comunicação se estava a efectuar correctamente carregou-se o programa na placa SPARTAN 3 e no botão de *reset* L14. De seguida variaram-se os *switches* de forma a visualizar cada uma das aquisições nos *displays* da placa. No entanto, apesar dos dois *displays* que representavam os 8 bits menos significativos da aquisição variarem visivelmente entre mudanças de *switches*, verificou-se que o terceiro não variava. Ligando a entrada do ADC a um sinal constante pode-se observar o mesmo, verificando-se que agora o ADC já não fornecia valores correspondentes à amostra analógica, mas valores muito mais baixos, mesmo com o sinal constante no máximo de 3.3V. Para se verificar se a comunicação se estava a efectuar correctamente, ligou-se a cada sinal da FPGA (*cs_FPGA*, *dout_FPGA*, *din_FPGA* e *clockSPI*) um canal do osciloscópio e observou-se o andamento dos sinais em questão. Verificou-se que de facto, os bits mais significativos da aquisição nunca ficavam a *high*, embora a comunicação se estivesse a efectuar correctamente, comportando-se os sinais *cs_FPGA*, *dout_FPGA* e *clockSPI* como o esperado. Por outro lado, no trabalho de laboratório anterior, o ADC tinha funcionado correctamente, tendo-se inclusivamente estabelecido uma recta de calibração cujos pontos experimentais se sobrepunham à recta teórica.

Depois de uma série de tentativas para resolver o problema, percebeu-se que provavelmente era o tempo entre aquisições sucessivas que era demasiado curto. De facto, existe uma curva de carga no sinal *din_FPGA* antes da definição da aquisição pelo ADC, e a rapidez das aquisições sucessivas não permitia que algum dispositivo no interior do ADC carregasse o suficiente para por os bits mais significativos da conversão analógica digital, os primeiros a serem transmitidos, a *high*.

Aquisição de um sinal variável com um ADC com interface SPI MPC3202

Este problema foi solucionado aumentando o tempo entre aquisições. Para tal alterou-se o código para em vez de esperar até que $count_{50kHz}$ fosse igual a 1 até iniciar as comunicações com o ADC, esperar que este sinal incrementasse até 1000, o que num contador de 50kHz corresponde a um tempo de espera de 0.02s. Os restantes pontos do programa que dependiam deste contador foram alterados em conformidade.

Uma vez carregada a nova versão do programa na placa, pode-se ver finalmente o último *display* a variar à medida que se aumentava a tensão constante à entrada do ADC. No entanto, os *displays* nunca mostraram o valor máximo, FFF para um sinal de *input* máximo de 3.3V. Provavelmente seria necessário aumentar ainda mais o tempo entre amostragens, mas considerou-se que esta alteração não era necessária, porque a gama de valores a amostrar variava entre 0 e 2V e para estes valores de tensão o ADC pareceu funcionar bem, de acordo com alguns cálculos rápidos feitos durante o laboratório. É de salientar que este problema consumiu algum tempo, pois demorou-se a perceber qual seria a solução até porque na *datasheet* do ADC utilizado não só não era feita qualquer referência à necessidade de tempos longos entre aquisições como também no próprio diagrama temporal do funcionamento do dispositivo se apresentavam aquisições sucessivas com tempos de permeio muito mais baixos que os que se verificaram ser efectivamente necessários.

Assim, constatou-se finalmente o bom funcionamento do programa para a gama de tensões a converter pelo ADC. Uma vez ultrapassada a fase da implementação e montagem do projecto pretendia-se, como foi referido no início deste trabalho, recolher 32 amostras do sinal em questão com intervalos entre amostras tais que a frequência de aquisição fosse inferior a duas vezes a frequência do sinal, superior e igual à mesma. Pretendia-se com este procedimento verificar a validade do critério de Nyquist, ao mesmo tempo que se punha novamente em prática a comunicação com o ADC e se constatava o seu bom funcionamento. Por outro lado, como o tempo entre amostragens estabelecido nesta fase do projecto era de 0.02s, pode-se concluir, desprezando o tempo de comunicação com o ADC, que para adquirir 32 amostras deveria passar um tempo de cerca de 0.64s. Assim, para se amostrarem relativamente poucos períodos do sinal sinusoidal, estabeleceu-se que este sinal deveria ter uma frequência de cerca de 3.8Hz, aproximadamente equivalente a um período de 0.250s, pelo que cerca de três períodos correspondem a 0.75s, um valor razoavelmente próximo de 0.64s. É evidente que uma vez aumentada ou diminuída a frequência de aquisição o número de períodos amostrados vai variar, mas pretendia-se apenas ter uma ideia da ordem de grandeza do mesmo. O sinal analógico a amostrar é representado na **Figura 3**.

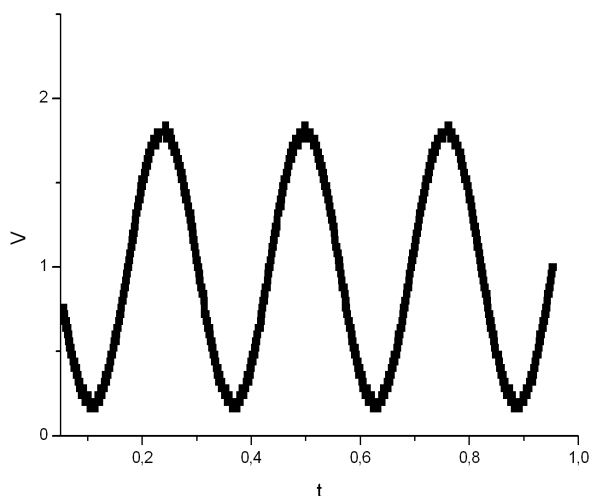


Figura 3 – Sinal analógico a amostrar

Assim, e porque o sinal do gerador de sinais escolhido tinha uma frequência de 3.8Hz, escolheu-se fazer uma amostragem de frequência 5Hz, outra de frequência 7.6Hz e outra de frequência 20Hz, correspondentes ao que era pedido. Para um melhor estudo da frequência de aquisição, ainda se fez uma amostragem de 40Hz, um valor muito superior à frequência postulada pelo critério de Nyquist. Assim, o programa foi mais uma vez alterado em conformidade e carregado para a placa em cada um dos casos, tendo-se recolhido os dados da aquisição provenientes do ADC através da sua visualização nos *displays* por manipulação dos *switches*. O tempo entre aquisições sucessivas foi controlado através da visualização do sinal ω_{FPGA} no osciloscópio, verificando-se que estava a ser cumprido pelo programa

Os gráficos obtidos com base nos valores recolhidos para cada uma das frequências de amostragem encontram-se nas **Figuras 4, 5 6 e 7**. Para uma melhor análise dos resultados obtidos, apresentam-se desde já os gráficos com os pontos unidos. É importante ter em conta que não se sabia o tempo correspondente a cada ponto experimental retirado, nem isso era fundamental para o trabalho em curso. Desta forma, optou-se por colocar todos os pontos iniciais em $t=0s$ e os restantes distanciados periodicamente, de acordo com a frequência de aquisição de cada um.

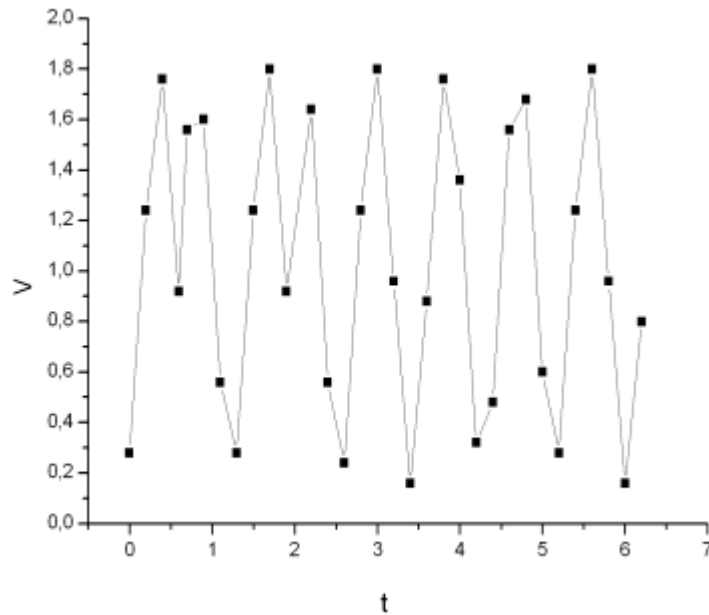


Figura 4 – Resultados experimentais para $f_{\text{aquisição}} (5\text{Hz}) \ll 2f_{\text{sinal}}$

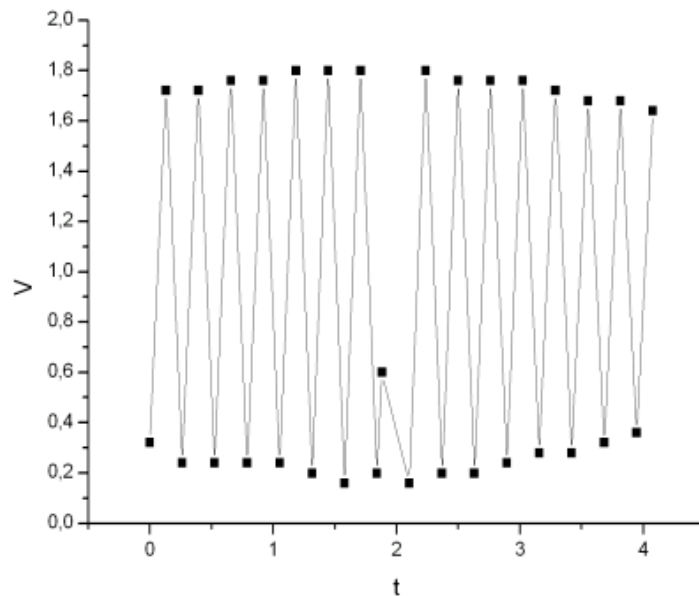


Figura 5 – Resultados experimentais para $f_{\text{aquisição}} (7.6\text{Hz}) \approx 2f_{\text{sinal}}$

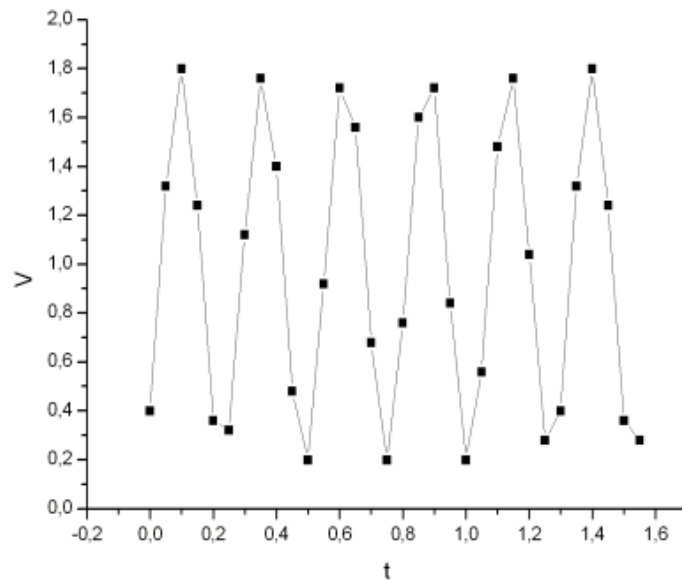


Figura 6 – Resultados experimentais para $f_{\text{aquisição}} (20\text{Hz}) > 2f_{\text{sinal}}$

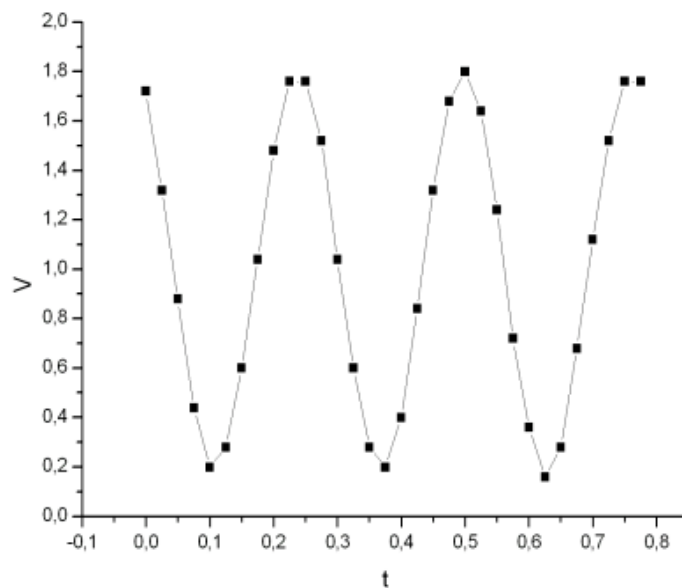


Figura 7 – Resultados experimentais para $f_{\text{aquisição}} (40\text{Hz}) \gg 2f_{\text{sinal}}$

Como seria de esperar, obtém-se uma má reconstrução do sinal para uma frequência de aquisição muito inferior à postulada pelo critério de Nyquist. Por outro lado a qualidade da reconstrução aumenta um pouco para uma frequência de aquisição que obedece a este critério, percebendo-se já o carácter periódico, de uma só frequência, do sinal. Este carácter fica mais claro para uma frequência de 20 Hz, onde a natureza sinusoidal do sinal já é mais perceptível. Contudo, qualquer uma destas aquisições não permite uma clara percepção do sinal original adquirido. É de salientar que o critério de Nyquist pode ser aplicado nestas circunstâncias, porque o sinal pode ser completamente reconstruído com a informação disponível recorrendo a métodos matemáticos.

No entanto, para as reconstruções efectuadas neste trabalho, apenas mediante a junção dos pontos da aquisição, a única aquisição em que se compreende claramente a característica sinusoidal do sinal é para 40 Hz, um valor muito superior à frequência postulada pelo critério de Nyquist. Como também seria de esperar, e porque o número de aquisições é constante igual a 32, visualiza-se um comportamento do sinal por menores períodos de tempo para frequências de aquisição crescentes. Para se reconstruir uma extensão temporal maior de sinal seria necessário aumentar a capacidade de memória do programa para adquirir mais pontos ou ter tentado fazer mais do que uma aquisição de 32 pontos com a mesma frequência de aquisição. No entanto, este procedimento não era essencial, porque foi possível reconstruir mais do que um período do sinal analógico, pelo que a sua natureza é compreendida com clareza para as frequências escolhidas que ultrapassam de forma significativa o critério de Nyquist.

De acordo com as especificações do enunciado foi ainda possível calcular a transformada FFT das sequências de amostras recolhidas para cada frequência de aquisição. Foi também feita uma transformada do sinal analógico de entrada do ADC. Os resultados encontram-se nas **Figuras 8, 9, 10, 11 e 12**.

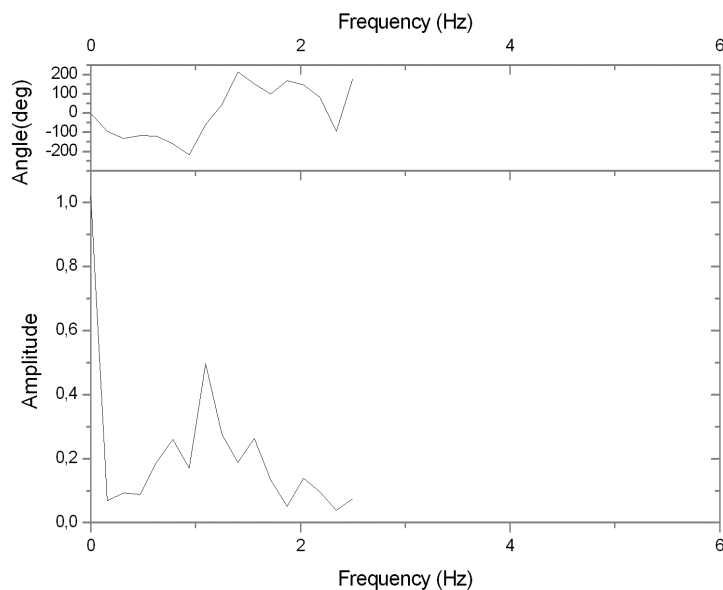


Figura 8 – Transformada FFT dos resultados experimentais para $f_{\text{aquisição}} (5\text{Hz}) \ll 2f_{\text{sinal}}$

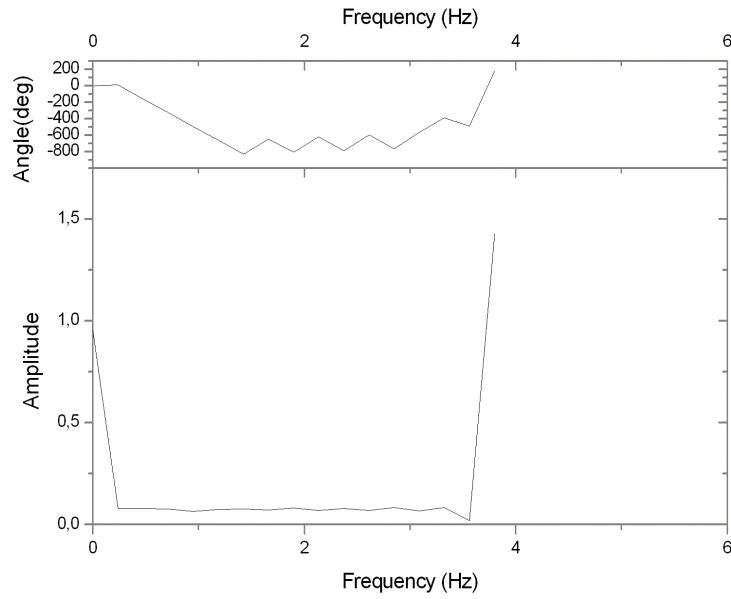


Figura 9 – Transformada FFT dos resultados experimentais para $f_{\text{aquisição}} (7.6\text{Hz}) \approx 2f_{\text{sinal}}$

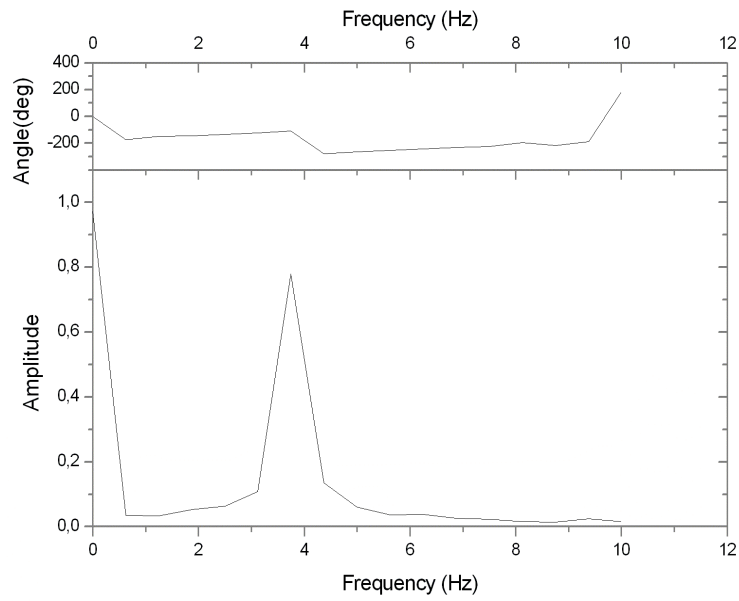


Figura 10 – Transformada FFT dos resultados experimentais para $f_{\text{aquisição}} (20\text{Hz}) > 2f_{\text{sinal}}$

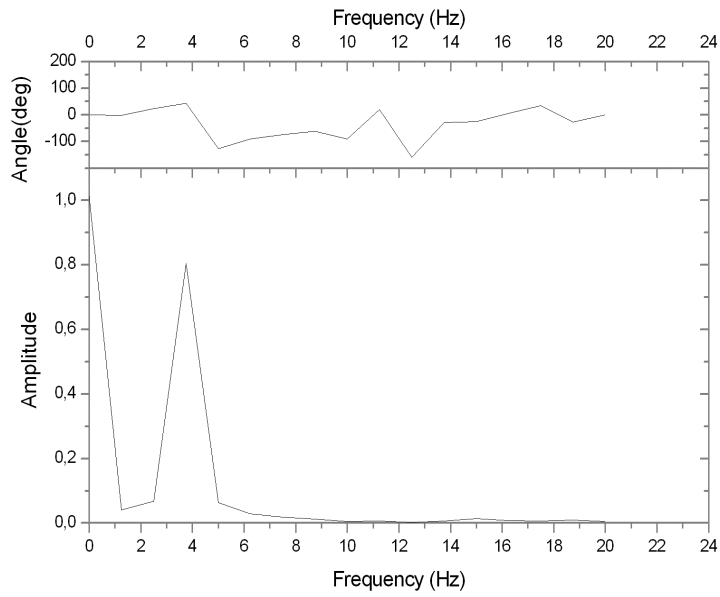


Figura 11 – Transformada FFT dos resultados experimentais para $f_{\text{aquisição}} (40\text{Hz}) \gg 2f_{\text{sinal}}$

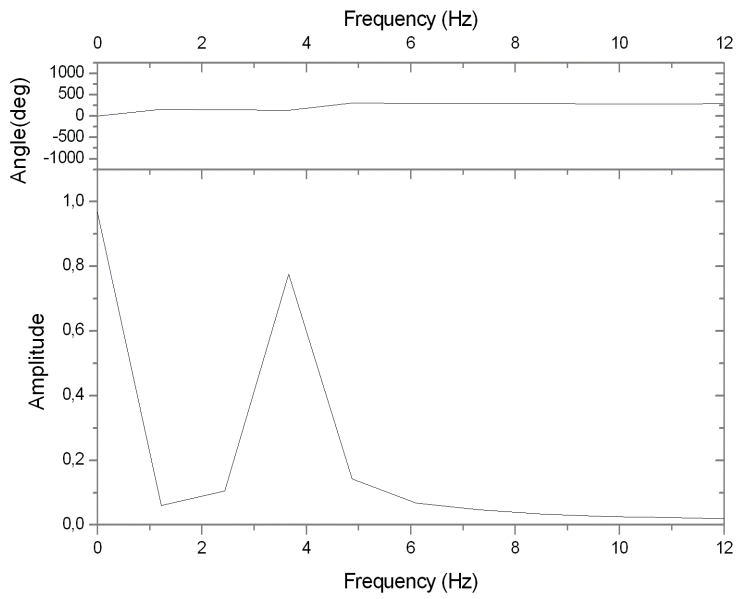


Figura 12 – Transformada FFT do sinal analógico

Através da análise da transformada FFT das diversas aquisições, observa-se que a partir da frequência de Nyquist é possível reconhecer qual a frequência do sinal, através do pico de amplitude bem definido à volta de 4Hz. Pode-se observar que a transformada FFT do sinal analógico (**Figura 12**) também possui um pico nesta frequência, como seria de esperar, porque foi assim que o sinal foi definido no gerador de sinais. No gráfico da **Figura 6**, correspondente à frequência de aquisição de 5Hz, não é possível visualizar um pico distinto, o que reforça a ideia de que o facto desta aquisição não obedecer ao critério de Nyquist não permite obter bons resultados, porque não se dispõe de informação suficiente para se reconstruir o sinal.

Conclusões

Este trabalho permitiu a aquisição de sinais analógicos variáveis utilizando um ADC, e recorrendo ao protocolo SPI. O trabalho efectuado baseou-se no laboratório anterior, tendo sido alterado apenas o necessário de forma a se obterem 32 aquisições para diferentes frequências. Mais uma vez, foi aprofundado o conhecimento da programação em VHDL, tendo o código sido adaptado para que, usando os *switches* da placa, os diferentes valores adquiridos fossem visualizados nos *displays* de 7 segmentos.

Foram obtidas aquisições para quatro frequências diferentes. Verificou-se que à medida que a frequência de aquisição aumenta, a qualidade do sinal adquirido também aumenta. Apesar da única aquisição que espelha de forma razoável as características do sinal analógico, possuir um valor de frequência de aquisição muito superior à postulada pelo critério de Nyquist, sabe-se que é possível reconstruir o sinal, recorrendo a métodos matemáticos que foram estudados e implementados durante as aulas, se o critério for cumprido. O facto da aquisição que cumpre o requerido pelo teorema de Nyquist fornecer uma FFT com um pico centrado nos 3.8Hz é uma indicação disto mesmo.

Por causa do problema verificado durante a aquisição inicial, percebeu-se que a implementação de um tempo relativamente longo entre cada aquisição é essencial, tendo sido necessário um aumento significativo no tempo entre aquisições para que os valores obtidos correspondam à recta de calibração determinada anteriormente.

Ainda foram efectuadas transformadas FFT dos sinais, que permitiram um estudo adicional das diversas aquisições, verificando-se que a aquisição de 5Hz (única que não respeitava o critério de Nyquist) é a única em que não se consegue obter um pico de amplitude para a frequência de 3.8Hz bem definido. Todos os restantes, incluindo a aquisição correspondente à frequência de Nyquist, possuem um pico bem definido próximo dos 4 Hz, a frequência do sinal analógico de entrada.

Bibliografia

- Manual do kit SPARTAN 3;
- datasheet* do integrado ADC MCP 3202;
- código e relatório do trabalho de laboratório II.