# Projecto de Sistemas Digitais I

Lic. em Engenharia de Sistemas e Informática

2º Ano

2006/2007

**António J. Esteves**

Dep. Informática, Universidade do Minho

Braga, Portugal

16 de Abril de 2007

# Projecto

**Objectivos**

*Com o presente projecto pretende-se que os alunos ponham em prática as competências adquiridas na disciplina de Sistemas Digitais I, nomeadamente: interpretar um enunciado e transformá-lo numa especificação em linguagem VHDL; utilizar circuitos combinacionais e sequenciais para resolver um problema; simular um circuito em VHDL; implementar (opcionalmente) um circuito num dispositivo de lógica programável (CPLD).*

## 1. Descrição do projecto

O problema proposto consiste em desenvolver uma pequena calculadora de 4 bits, que executa algumas instruções básicas e funciona em torno de uma pilha com 4 posições. Para tal deve-se analisar o problema, obter o diagrama de blocos, descrever e simular em VHDL e implementar (opcionalmente) a solução obtida na CPLD do PG04 v2 [1].

A calculadora a realizar executa instruções orientadas à pilha, com 0 ou 1 operandos, em que o código das instruções ocupa 3 bits e o operando 4 bits (figura 1).

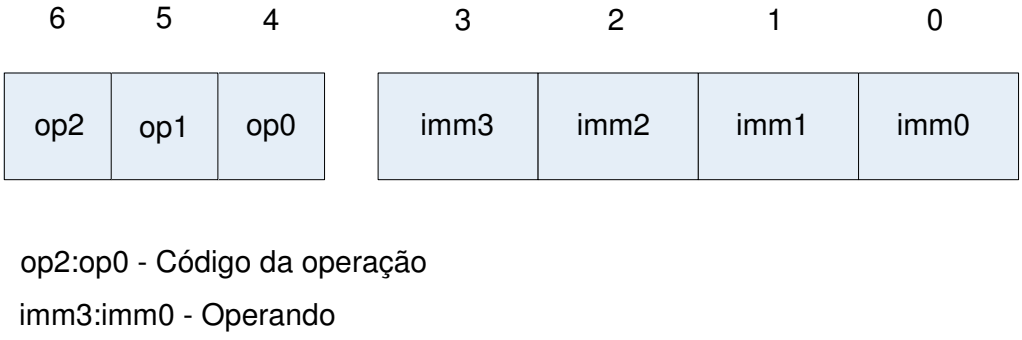


Figura 1: Formato das instruções a executar pela calculadora.

As instruções que a calculadora executa são as que se apresentam na tabela 1. O eventual operando de uma instrução está em complemento para 2, ou seja, varia de -8 a +7.

[1] O diagrama de blocos do PG04 v2 é apresentado na figura 20 do anexo A do guia tutorial das aulas práticas.

| *Sintaxe* | *Explicação da funcionalidade* | *Flags afectadas* |
|---|---|---|
| PUSH imm4 | SP=SP+1 ; [SP]=imm4 | |
| POP | SP=SP-1 | |
| ADD | T=[SP] ; SP=SP-1; [SP]= T + [SP] | Z, S, C |
| SUB | T=[SP] ; SP=SP-1; [SP]= T - [SP] | Z, S, C |
| ADDI imm4 | T=[SP] ; [SP]= T + imm4 | Z, S, C |
| SUBI imm4 | T=[SP] ; [SP]= T - imm4 | Z, S, C |
| NOT | T=[SP] ; [SP]= /T | |

Legenda:

| | |
|---|---|
| SP | - registo que aponta para a pilha |
| T | - registo auxiliar |
| [SP] | - conteúdo da posição da pilha apontada por SP |
| imm4 | - operando com 4 bits |
| Z, S, C | - flags zero, sinal, carry |
| / | - negação |

Tabela 1: Conjunto de instruções da calculadora.

### Notas sobre as flags

A flag de zero (Z) fica a 1 se o resultado da soma/subtracção for nulo, caso contrário fica a 0.

A flag de sinal (S) fica a 1 se o resultado da soma/subtracção for negativo, caso contrário fica a 0.

A flag de carry (C) fica a 1 se o resultado da soma/subtracção originar transporte no bit mais significativo, caso contrário fica a 0.

Por questões de simplificação do projecto, a operação *NOT* não afecta as flags.

### Notas sobre a arquitectura da calculadora

A arquitectura deve incluir uma pilha com profundidade 4 e palavras de 4 bits. A pilha funciona de forma circular, ou seja, quando se estiver na última (primeira) posição e for incrementado (decrementado) o SP, salta-se para a primeira (última) posição.

O conteúdo da posição da pilha apontada pelo registo SP deve ser disponibilizado para o exterior, de modo a poder ser visualizado num visor de 8-segmentos (7 segmentos + 1 ponto) do PG04 v2. Para distinguir os números positivos (0 a 7) dos negativos (-8 a -1), usa-se o ponto (".") do visor para desenhar os números negativos (figura 2).

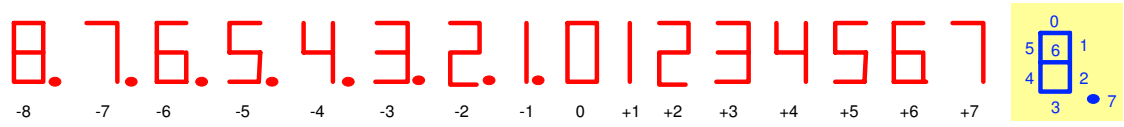


Figura 2: Aspecto gráfico a usar no visor de 8-segmentos, para mostrar o conteúdo da pilha.

As instruções, com o formato da figura 1, devem ser fornecidas uma-a-uma à calculadora a partir de 7 dos 8 interruptores do PG04 v2. O oitavo interruptor, a que vamos chamar ***execute***, deve ser utilizado para dar a ordem de execução da instrução presente nos restantes 7 interruptores.

As flags Z, S e C devem ser disponibilizadas para o exterior, de modo a poderem ser

visualizadas em LEDs do PG04 v2.

Para controlar a execução das instruções deve usar-se uma máquina de estados.

# 2. Tarefas a realizar

## 2.1 Diagrama de blocos da arquitectura

Após uma análise cuidada do enunciado do problema, obtenha o **diagrama de blocos** para a calculadora de 4 bits proposta. Comece por identificar as entradas e saídas do sistema e depois refine o interior do sistema com os blocos necessários.

Numa segunda fase, converta o diagrama de blocos para um **esquemático** detalhado que servirá de base para a posterior descrição em VHDL. Este esquemático deverá incluir componentes que constituem o caminho de dados e uma unidade de controlo (**máquina de estados**). Nesta fase deve atribuir a cada instrução o respectivo código, isto é, seleccionar a sequência binária correspondente aos bits op2..op0 do formato da instrução que a identificam univocamente. O principal critério a usar nesta selecção, é a minimização da quantidade de lógica necessária à implementação da calculadora.

Numa terceira fase, desenhe o **diagrama de estados** para a unidade de controlo, de modo a gerar os sinais necessários ao correcto funcionamento do caminho de dados.

## 2.2 Conversor de binário para 8-segmentos em VHDL

Assumindo que a arquitectura inclui um conversor de binário para 8-segmentos, que é utilizado para visualizar o conteúdo da pilha, represente numa tabela de verdade a funcionalidade desse conversor. Obtenha a expressão minimizada para cada uma das 8 saídas do conversor, usando mapas de Karnaugh.

Com base nas expressões minimizadas, descreva em VHDL (estilo fluxo de dados) a arquitectura e a entidade do conversor.

## 2.3 Descrever o caminho de dados em VHDL

Descreva em VHDL os componentes da arquitectura que constituem o caminho de dados. Sempre que possível, utilize o estilo comportamental. Por exemplo, se utilizar registos e multiplexadores eles devem ser descritos de forma comportamental. Para especificar a interligação entre estes componentes utilize o estilo estrutural.

## 2.4 Descrever a unidade de controlo em VHDL

Descreva em VHDL, estilo comportamental, o diagrama de estados obtido no ponto 2.1.

## 2.5 Simulação em VHDL

Fazendo uso do ambiente de simulação proporcionado pela ferramenta Active-HDL, verifique o funcionamento da calculadora descrita em VHDL. Para esse fim, é pedido que se escreva um *testbench* contendo pelo menos os seguintes cenários:

- Uma sequência de instruções PUSH para testar o conversor binário para 8 segmentos;
- Uma sequência de instruções PUSH que encha totalmente a pilha, seguida de uma sequência de POPs que a esvazie;
- Efectuar a operação "res = op1 - op2", em que o resultado seja zero;
- Efectuar a operação "res = op1 - op2", em que o resultado seja negativo;
- Efectuar a operação "res = op1 + op2", em que o resultado cabe na gama de representação da calculadora de 4 bits;
- Efectuar a operação "res = op1 + op2", em que o resultado ultrapassa a gama de representação da calculadora de 4 bits;
- Efectuar a operação "res = / op1";
- Efectuar a operação "res = (4 + 3) - 2", envolvendo as instruções ADDI e SUBI.

## 2.6 Implementação a partir de VHDL

**NOTA: A execução desta parte do projecto é opcional, mas vale 0,5 valores.**

Utilizando a ferramenta de desenvolvimento da Xilinx, sintetize o circuito usando a descrição VHDL obtida nos pontos 3 a 5 e posteriormente implemente-a com a CPLD do PG04 v2. Para sintetizar a descrição do circuito e programar a CPLD, recomendam-se os seguintes passos:

i. No ambiente de desenvolvimento da Xilinx, crie um novo projecto do tipo HDL usando as opções:
   - `Device family`: XC9500 CPLDs
   - `Device`: XC95108
   - `Package`: PC84
   - `Speed grade`: -15
   - `Top level module type`: HDL
   - `Synthesis tool`: XST (VHDL)

ii. Crie o projecto vazio e adicione *a posteriori* os ficheiros VHDL obtidos nos pontos 3 a 5;

| *Sinal* | *Pino da CPLD* | *Sinal* | *Pino da CPLD* |
|---|---|---|---|
| execute | P41 | result[7] | P4 |
| opcode[2] | P40 | result[6] | P11 |
| opcode[1] | P39 | result[5] | P7 |
| opcode[0] | P37 | result[4] | P1 |
| operand[3] | P36 | result[3] | P2 |
| operand[2] | P35 | result[2] | P3 |
| operand[1] | P34 | result[1] | P5 |
| operand[0] | P33 | result[0] | P6 |
| Zflag | P84 | reset | P74 |
| Sflag | P83 | clk | P9 |
| Cflag | P82 | | |

Tabela 2: Atribuição dos sinais de entrada e saída da CPLD aos pinos do encapsulamento PLCC84.

iii. Edite a localização desejada para os sinais de entrada e de saída do circuito quando implementado na CPLD. Para isso utiliza-se o utilitário `PACE`, que pode ser executado através da opção `Assign Package Pins`. Ao executar o `PACE` opte por responder `YES` à primeira pergunta colocada e atribua graficamente a localização dos sinais do controlador aos pinos da CPLD, usando o mapeamento da tabela 2.

   A atribuição processa-se arrastando cada sinal da lista `I/O pins` para o pino adequado no esquema do encapsulamento da CPLD, o qual se encontra na janela `Package pins for XC95108-PC84-15`. Guarde o ficheiro no final da atribuição e feche o utilitário `PACE`.

iv. Gere o ficheiro de configuração da CPLD através da opção `Generate Programming File`, mas para isso deve ter-se seleccionado/marcado a entidade (ficheiro) VHDL que descreve o circuito.

v. Inspeccione os resultados da síntese, sobre a forma de esquemático, através da opção `View RTL Schematic`. O resultado é o esperado, ou seja, a estrutura gerada pela síntese corresponde ao esquemático do ponto 3? Que conclui da forma como a máquina de estados foi sintetizada?

vi. Inspeccionando os relatórios de síntese, qual a frequência máxima que pode usar no relógio da implementação?

vii. Configure a CPLD do PG04 v2 usando os seguintes passos:

   - Coloque os *jumpers* JP6 e JP7 do PG04 v2 no estado ON, para se poder programar a CPLD pela porta e cabo paralelos, e confirme que o PG04 v2 está ligado ao PC através dum cabo paralelo;
   - Execute o utilitário `Xilinx JTAG Programmer`;
   - O `JTAG Programmer` deve detectar que existe na cadeia de JTAG, entre os sinais TDI e TDO, uma CPLD XC95108;
   - Clique duas vezes com o rato sobre o símbolo da CPLD presente na cadeia JTAG detectada e especifique o ficheiro de configuração JEDEC a enviar para a CPLD. Escolha o ficheiro JEDEC gerado no ponto (iv);
   - Clique uma vez com o rato sobre o símbolo da CPLD presente na cadeia JTAG, de modo a que este símbolo fique seleccionado, e em seguida seleccione o comando `Program` do menu `Operations`. Com a opção `Erase before`

`programming` seleccionada, mande programar a CPLD. Se não ocorrerem erros, a CPLD está apta a funcionar, implementando o circuito projectado.

Pode finalmente verificar o funcionamento do circuito, aplicando na sua entrada sequências de instruções análogas às usadas nos cenários da simulação. Para casa instrução testada, anote o valor aplicado na entrada e o resultado obtido no visor de 8-segmentos e nos LEDs.

# 3. Metodologia de avaliação

A nota final do projecto resultará de duas componentes:

- A avaliação contínua, que será feita pelo docente durante 4 aulas práticas, a partir do início do mês de Maio, e avaliará a evolução e conclusão do projecto. Esta componente vale 2 valores.
- A avaliação final, que será feita pelo docente das práticas na última aula prática do semestre, e avaliará os resultados obtidos com o projecto. Esta componente vale 3 valores.
- Em cada sessão de avaliação, contínua e final, os alunos devem entregar por escrito os elementos relativos aos items em avaliação (e.g., diagramas, esquemáticos, mapas de Karnaugh, ...).

# 4. Calendarização

Para atingir um bom desempenho na realização do projecto, medido pela componente de avaliação contínua, os alunos devem respeitar a seguinte calendarização:

- Na 1ª semana será avaliado o diagrama de blocos e o esquemático.
- Na 2ª semana será avaliado o diagrama de estados e o conversor de binário para 8-segmentos.
- Na 3ª semana será avaliada a arquitectura da calculadora em VHDL.
- Na 4ª semana será avaliada a simulação em VHDL e a globalidade do projecto.

# 5. Regras de Funcionamento

- Qualquer tentativa de fraude será penalizada com a atribuição de zero valores na componente de projecto a todos os elementos do grupo.
- A não comparência de um ou mais elementos do grupo em cada sessão de avaliação resultará, por defeito, na atribuição de zero valores aos elementos faltosos na componente em avaliação. Justificando a ausência no prazo de uma semana e sendo possível marcar uma sessão extra no horário de atendimento do respectivo docente, esta avaliação poderá ser revista.