Projeto de circuito

MAC0329 – Álgebra booleana e aplicações (DCC / IME-USP — 2019)

– Todas as etapas do projeto deverão ser feitas com o *Logisim* (http://www.cburch.com/logisim/) –

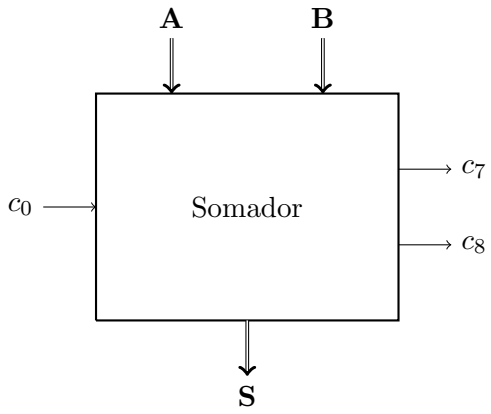
– O projeto poderá ser feito em grupo de até três pessoas –

Parte 1: Circuito Somador – entrega no PACA, até 31/03

Nesta primeira etapa do projeto (Parte 1), o objetivo é a implementação de um circuito somador de 8 *bits*, e sua utilização para realizar as operações de adição e subtração com números de 8 *bits*.

1 Especificação de um somador de 8 *bits*

Em nosso projeto, o somador de 8 bits terá a seguinte configuração



Entradas:

A $= (a_7, a_6, a_5, a_4, a_3, a_2, a_1, a_0)$ (operando 1)
B $= (b_7, b_6, b_5, b_4, b_3, b_2, b_1, b_0)$ (operando 2)
c_0 (*carry* na coluna 0)

Saídas:

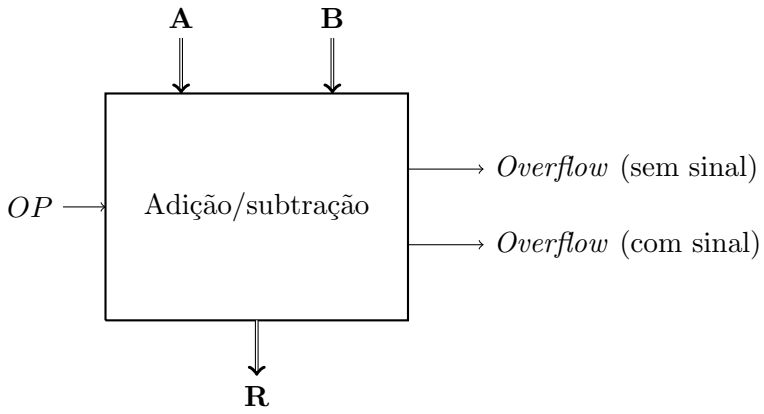
S $=$ **A** $+$ **B** (8 *bits* da soma)
c_7 (*carry* na coluna 7)
c_8 (*carry* na coluna 8)

2 Modularização

Modularização relaciona-se com organização. No caso de circuitos lógicos, um circuito complexo pode ser decomposto em termos de combinação de subcircuitos. Cada subcircuito pode ser visto como uma "caixa-preta" e deve ter uma interface e funcionalidade bem definidas. Com isso, conseguimos organizar um circuito complexo de forma compacta, o que facilita o seu entendimento, mesmo que não saibamos exatamente como cada subcircuito está implementado. Além disso, um subcircuito pode ser utilizado em diversas partes de um circuito maior, evitando repetições.

Você deverá organizar o circuito do EP1 em três circuitos/subcircuitos:

1. `main` : o circuito principal, que servirá para testar o somador. Aqui espera-se um circuito com a seguinte interface:



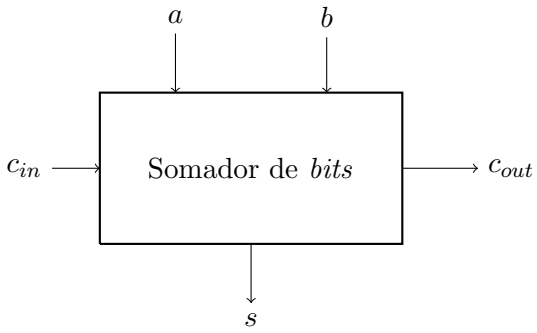
Neste diagrama,

- **A** e **B** são os operandos, ambos de 8 *bits*;
- *OP* é um *bit* para indicar o tipo de operação: se 0, deve ser calculada a adição $\mathbf{A} + \mathbf{B}$ e se 1 deve ser calculada a subtração $\mathbf{A} - \mathbf{B}$;
- **R** é o resultado da operação (8 *bits*);
- "*overflow* sem sinal" é 1 *bit*: se 1, indica que a operação calculada, interpretando-se os números como sem sinal, resultou em *overflow*;
- "*overflow* com sinal" é 1 *bit*: se 1, indica que a operação calculada, interpretando-se os números como com sinal, resultou em *overflow*.

O circuito principal deve utilizar o circuito `somador`, descrito no próximo item.

2. `somador`: este deve ser o circuito somador descrito na seção 1 acima e ele deve ser construído usando o subcircuito `somabits` especificado a seguir.

3. `somabits`: este deve ser o circuito somador de *bits*, visto em sala de aula. Ele recebe três *bits*, a, b e c_{in}, e devolve dois *bits*, s e c_{out} (veja capítulo 3 das notas de aula). O diagrama a seguir mostra como deve ser organizado este circuito:



Observação: Embora aqui estejamos falando de modularização de circuitos lógicos, esse conceito estará bastante presente na especificação e desenvolvimento de softwares em geral. Em MAC0110, em breve vocês aprenderão "funções", que correspondem aos elementos modulares na organização do código de um programa.

3 O que fazer e entregar

O *Logisim* permite a criação de vários subcircuitos, que podem ser gravados em um único arquivo.

Vocês deverão gerar um arquivo `somador.circ` contendo os três circuitos descritos acima. Ao entrar no *Logisim*, há um painel principal que corresponde ao `main`, no qual pode ser criado o circuito principal. Utilizando-se a opção `Adicionar subcircuito ...` do menu `Projeto`, pode-se criar outros painéis, para os demais subcircuitos.

A ordem natural para o EP1 é fazer o circuito seguindo a estratégia *bottom-up*, isto é, começa-se com o subcircuito mais básico (`somabits`), em seguida faz-se o `somador`, e finalmente o `main`. Cada um deles deve ser testado individualmente, antes de ser empregado em um circuito de nível superior.

Poderão ser utilizados

- pinos (entrada e saída)
- portas lógicas
- *buffers* e inversores (controlados ou não)
- *splitters*

Observação: O *template* fornecido `somador.circ` já inclui os painéis para os circuitos `main` e `somador`, cada um contendo os pinos de entrada e saída dos respectivos circuitos. Vocês podem utilizar este arquivo como ponto de partida para o seu EP1. Note, especialmente, o uso de *bits* largos juntamente com os *splitters* e também a definição de rótulos/nomes para os diferentes componentes do circuito (estes são úteis para documentação/facilitar a leitura).

Entregar no PACA o arquivo `somador.circ`, gerado pelo *Logisim*, contendo os três circuitos especificados acima. Se o projeto for realizado em grupo, apenas um dos membros deve fazer a entrega (não esqueça de incluir o nome de todos os membros do grupo na documentação no próprio circuito).

4 Critérios de avaliação

Serão avaliados os seguintes aspectos:

- corretude dos circuitos
- aderência à organização proposta, incluindo a localização dos pinos de entrada e saída
- documentação: atribuição de rótulos aos pinos de entrada e saída, outros nomes/descrições/comentários sucintos úteis para o entendimento dos circuitos, identificação da tarefa e dos autores.
- cumprimento do prazo