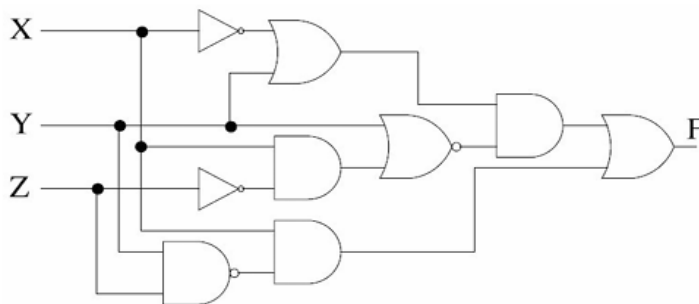


Fórmulas Booleanas Derivadas dos Circuitos

Daniel Marchioro, Inácio L. S. Viana, Sarah C. A. Pereira.
DCC301 - A0C

Após a análise dos circuitos requeridos, obtivemos as seguintes fórmulas booleanas:

(A)



- $F = ((\neg X \wedge Y) \wedge \neg(Y \vee (X \wedge \neg Z))) \vee (X \vee \neg(Y \wedge Z))$

(B) Somador completo de 8 bits:

Para um circuito somador completo, temos dois sinais de saída: o resultado da soma e o Carry Out:

- $S = A \oplus B \oplus \text{Carry In}$
- $\text{Carry Out} = (A \wedge B) \vee (\text{Carry In} \wedge (A \vee B))$

A modelagem de um circuito somador de 8 bits segue uma lógica em cascata, onde o Carry In sempre vai receber o Carry Out do somador anterior:

- $S(n) = A(n) \oplus B(n) \oplus \text{Carry In}(n)$
- $\text{Carry Out}(n) = (A(n) \wedge B(n)) \vee (\text{Carry In}(n) \wedge (A(n) \vee B(n)))$

(C) Unidade de controle (codificada em uma Máquina de Estados Finitos Completa) do processador multiciclo MIPS:

Os sinais de saída são definidos como as opções de controle oferecidas pelo processador MIPS, são eles:

- $\text{Jump} = \neg S2 \wedge S1 \wedge \neg S0 \wedge \neg \text{Op5} \wedge \neg \text{Op4} \wedge \neg \text{Op3} \wedge \neg \text{Op2} \wedge \text{Op1} \wedge \text{Op0};$
- $\text{ALUOp0} = \neg S2 \wedge S1 \wedge \neg S0 \wedge \neg \text{Op5} \wedge \neg \text{Op4} \wedge \neg \text{Op3} \wedge \neg \text{Op2} \wedge \text{Op1} \wedge \neg \text{Op0};$
- $\text{ALUOp1} = \neg S2 \wedge S1 \wedge \neg S0 \wedge \neg \text{Op5} \wedge \neg \text{Op4} \wedge \neg \text{Op3} \wedge \neg \text{Op2} \wedge \neg \text{Op1} \wedge \neg \text{Op0};$
- $\text{Branch} = \neg S2 \wedge S1 \wedge \neg S0 \wedge \neg \text{Op5} \wedge \neg \text{Op4} \wedge \neg \text{Op3} \wedge \neg \text{Op2} \wedge \text{Op1} \wedge \neg \text{Op0};$
- $\text{MemWrite} = \neg S2 \wedge \neg S1 \wedge S0 \wedge \neg \text{Op5} \wedge \neg \text{Op4} \wedge \text{Op3} \wedge \neg \text{Op2} \wedge \neg \text{Op1} \wedge \neg \text{Op0};$

- $\text{MemRead} = (\neg S2 \wedge \neg S1 \wedge \neg S0) \vee (\neg S2 \wedge \neg S1 \wedge S0 \wedge \neg \text{Op5} \wedge \neg \text{Op4} \wedge \neg \text{Op3} \wedge \text{Op2} \wedge \text{Op1} \wedge \text{Op0});$
- $\text{RegWrite} = S2 \wedge \neg S1 \wedge \neg S0 \wedge (\neg \text{Op5} \wedge \neg \text{Op4} \wedge \neg \text{Op3} \wedge \neg \text{Op2} \wedge \neg \text{Op1} \wedge \neg \text{Op0} \vee \neg \text{Op5} \wedge \neg \text{Op4} \wedge \neg \text{Op3} \wedge \text{Op2} \wedge \text{Op1} \wedge \text{Op0} \vee \neg \text{Op5} \wedge \neg \text{Op4} \wedge \text{Op3} \wedge \neg \text{Op2} \wedge \neg \text{Op1} \wedge \neg \text{Op0});$
- $\text{MemtoReg} = \neg S2 \wedge \neg S1 \wedge S0 \wedge \neg \text{Op5} \wedge \neg \text{Op4} \wedge \neg \text{Op3} \wedge \text{Op2} \wedge \text{Op1} \wedge \text{Op0};$
- $\text{ALUSrc} = \neg S2 \wedge S1 \wedge \neg S0 \wedge (\neg \text{Op5} \wedge \neg \text{Op4} \wedge \neg \text{Op3} \wedge \text{Op2} \wedge \text{Op1} \wedge \text{Op0} \vee \neg \text{Op5} \wedge \neg \text{Op4} \wedge \text{Op3} \wedge \neg \text{Op2} \wedge \neg \text{Op1} \wedge \neg \text{Op0});$
- $\text{RegDst} = \neg S2 \wedge S1 \wedge \neg S0 \wedge \neg \text{Op5} \wedge \neg \text{Op4} \wedge \neg \text{Op3} \wedge \neg \text{Op2} \wedge \neg \text{Op1} \wedge \neg \text{Op0}.$

(D) $F = ABCD + ABC'D + ABC'D' + AB'CD + A'BCD + A'BCD' + A'BC'D + A'B'C'D:$

- $F = (A \wedge B \wedge C \wedge D) \vee (A \wedge B \wedge \neg C \wedge D) \vee (A \wedge B \wedge \neg C \wedge \neg D) \vee (A \wedge \neg B \wedge C \wedge D) \vee (\neg A \wedge B \wedge C \wedge D) \vee (\neg A \wedge B \wedge C \wedge \neg D) \vee (\neg A \wedge B \wedge \neg C \wedge D) \vee (\neg A \wedge \neg B \wedge \neg C \wedge D);$

(E) Unidade Lógica e Aritmética (ULA) de 8 bits com as funções de subtração, XOR, NAND, NOR e shift de 2 bits à esquerda:

Cinco tipos de saídas serão definidas, subtração, XOR, NAND, NOR e shift de 2 bits à esquerda:

- $\text{XOR}(n) = (A(n) \wedge \neg B(n)) \vee (\neg A(n) \wedge B(n));$
- $\text{NAND}(n) = \neg(A(n) \wedge B(n));$
- $\text{NOR} = \neg(A(n) \vee B(n));$

Para o subtrator de 8 bits, ele segue uma lógica similar do somador, ele possui duas saídas, a subtração e o Borrow Out e seu funcionamento também se dá em cascata:

- $S(n) = A(n) \oplus B(n) \oplus \text{Borrow In}(n);$
- $\text{Borrow Out}(n) = (\neg A(n) \wedge B) \vee (\neg A(n) \wedge \text{Borrow In}(n)) \vee (B \wedge \text{Borrow In}(n));$

A operação de shift à esquerda, não pode ser representada logicamente por meio de fórmulas booleanas.

(F) Fluxo de execução de uma instrução do tipo R no MIPS de 32bits.

- Instrução do tipo R = $\text{RegDst} \wedge \neg \text{ALUSrc} \wedge \neg \text{MemtoReg} \wedge \text{RegWrite} \wedge \neg \text{MemRead} \wedge \neg \text{MemWrite}.$