Universität des Saarlandes
FR 6.2 - Informatik
Prof. Dr. W.J. Paul
Dipl.-Ing. Christoph Baumann

Exercise Sheet 2
Computer Architecture II
(Due: Nov 5th, 2013)

Exercise 1: (4/2 Adder) **(3+3+3 Points)**

A 4/2-Adder is a circuit with inputs $a, b, c, d \in \mathbb{B}^n$ and outputs $s, t \in \mathbb{B}^n$, fulfilling the specification:

$$\langle s \rangle + \langle t \rangle \;\equiv\; \langle a \rangle + \langle b \rangle + \langle c \rangle + \langle d \rangle \qquad \mod 2^n$$

Construct an implementation using two 3/2-Adders and prove its correctness! How can we compute strings s', t' such that $\langle s' \rangle + \langle t' \rangle = \langle a \rangle + \langle b \rangle + \langle c \rangle + \langle d \rangle$? Prove your answer!

Exercise 2: (Excess Full Adders) **(2+3+1 Points)**

The multiplication of binary numbers $a \in \mathbb{B}^n, b \in \mathbb{B}^m$ is implemented efficiently by summing up the partial products $\langle a \rangle \cdot b_i \cdot 2^i$ in a so-called *Wallace tree* consisting of 4/2 and 3/2 adders. From the leaves to the root of the tree the width of the intermediate results is growing, therefore we need to use wider 4/2 adders and consequently also more full adders. We denote as *excess full adders* the additional full adders needed compared to a tree of n-bit 4/2-adders. In the lecture we have shown that for arbitrary m the number of excess full adders has an upper bound of $\mu \cdot m$, where $\mu = \log(M/4)$, $M = 2^{\lceil \log m \rceil}$, and $\frac{3}{4}M \leq m \leq M$. For $m = 2^k$ we can conduct a precise analysis, taking the following steps:

a) How many levels of 4/2-adders does the Wallace tree have? How many 4/2-adders are in each level?

b) How many excess full adders do we need for a single 4/2-adder node in a given level l? What is the total number of excess full adders for the Wallace tree?

c) Compare your result with the original estimate for $m = 2^k$!

Exercise 3: (Booth Recoding: Implementation) **(8 Points)**

Provide an efficient circuit calculating the following predicates:

$$(|B_{2j}| = 1) \qquad\qquad (|B_{2j}| = 2) \qquad\qquad s_{2j}$$

Exercise 4: (Booth Recoding: Correctness Proof) **(5 Points)**

In the lecture we proved for $j > 0$ that the binary representation e_{2j} of E_{2j} can be computed as:

$$\langle e_{2j} \rangle \;=\; \begin{cases} \langle s_{2j}, d_{2j} \oplus s_{2j} \rangle + s_{2j} & : \quad j > 0 \\ \langle \overline{s_0} s_0 s_0, d_0 \oplus s_0 \rangle + s_0 & : \quad j = 0 \end{cases}$$

Prove the claim for $j = 0$!

Exercise 5: (Booth Recoding: Cost Analysis) **(5+5+2 Points)**

We have said that due to reduced wire delays in the physical layout of the Wallace tree, booth recoding speeds up multiplication by a constant factor. Now we want to calculate the cost of booth recoding. For simplicity we can assume $m = 2^k$.

- Calculate the number of full adders needed for an n, m-bit Wallace tree with out booth recoding!
- Calculate the same number for a Wallace tree with booth recoding!
- Compare your results! How do the costs of booth recoding compete against the original cost for big m and n?