Saarland University
Department 6.2 – Computer Science
Dr. M. A. Hillebrand
Prof. Dr. W. J. Paul

Computer Architecture I – WS 06/07

Exercise Sheet 9 **(due: 15.01.07)**

Exercise 1: (extended encoder) **(10+10=20 points)**

In the lecture we have seen that the *hit* signal is computed as $\bigvee_i hit_i$. On the last exercise sheet, we have seen that this computation can be performed in a fast way using an parallel prefix computation. In this exercise we show a possibility to compute the *hit* signal through an extended encoder.

An encoder is a circuit with inputs $x[2^n - 1 : 0]$ and outputs $y[n-1:0]$such that:

$$unary(a) \Rightarrow \langle b \rangle = \langle a \rangle_{\mathrm{u}}$$

We abbreviate this as enc_{2^n}. An extended encoder is a circuit with inputs $x[2^n - 1 : 0]$ and outputs $y[n:0]$. It is recursively defined by:

$$\begin{aligned} encf_{2^1}(a) &:= (a[1] \vee a[0]) \circ a[1] \\ encf_{2^{n+1}}(a) &:= (lo[n] \vee hi[n]) \circ comp\_res(n, n, lo, hi) \\ \text{with } lo &:= encf_{2^n}(a[2^n - 1 : 0]) \\ \text{and } hi &:= encf_{2^n}(a[2^{n+1} - 1 : 2^n]) \end{aligned}$$

Where *comp_res* is recursively defined by:

$$\begin{aligned} comp\_res(0, n, lo, hi) &:= \begin{cases} hi[0] & : n = 0 \\ lo[0] \vee hi[0] & : otherwise \end{cases} \\ comp\_res(i, n, lo, hi) &:= \left(\begin{cases} hi[n] & i = n \\ lo[i] \vee hi[i] & : otherwise \end{cases} \right) \circ comp\_res((i-1), n, lo, hi) \end{aligned}$$

You have to prove:

1. $\forall a \in \mathbb{B}^{2^n}.\ encf_{2^n}(a)[n] = or_{2^n}(a)$
2. $\forall a \in \mathbb{B}^{2^n}.\ encf_{2^n}(a)[n-:0] = enc_{2^n}(a)$

Exercise 2: (SPR) **(10 points)**

In this exercise you have to construct the SPR register. In the lecture you have seen all the update functions for the data stored in the SPR.

Recall that the SPR is read in three situations:

- $movs2i$ reads from $SPR(sa(c))$
- SR is read in the memory stage
- *rfe* instruction reads the two exception PCs

Moreover, the SPR is also updated in three situations

- $movi2s$ writes $SPR(sa(c))$
- SR is updated by the *rfe* instruction
- all special purpose registers are updated by *JISR*