University of the Saarland
Department 6.2 - Informatik
Prof. Dr. W.J. Paul

Computer Architecture I - WS 03/04

(due: 10.12.2003)

Exercise 1: (closed formulas) **(6 points)**

Derive closed formulas (without recursion or $\sum$ symbols) for parameters of the decoder layout considered in the lecture[1]. So, you need to give the following formulas[2]:

- height $h(dec(n))$
- width $w(dec(n))$
- load $load(x_i, n)$ for inputs x_i
- time $time(y_i, n)$ for outputs y_i (you can give the upper bound in the same way as in the lecture)

Exercise 2: (CLS layout) **(8 points)**

Perform analysis for the layout of the n-bit cyclic left shifter considered in the lecture. It takes $a[n-1:0]$ and $b[\log n - 1:0]$ as inputs and provides outputs $r[n-1:0]$, where $r = cls(a, \langle b \rangle)$. Compute values of:

- height $h(cls(n))$
- width $w(cls(n))$
- load $load(a_i, n)$ for inputs a_i and load $load(b_j, n)$ for inputs b_j
- time $time(r_i, n)$ for outputs r_i

Exercise 3: ((de)coding DLX instructions) **(2+2 points)**

The assembler instructions are:

- r-type: Mnemonic / destination register / source register 1 / source register 2

  Example: add R1 R2 R3 (adds the value of register R2 and R3 and stores the result in register R1)

- i-type: Mnemonic / destination register / source register / 16 bit constant

  Example: addi R1 R2 523

- j-type: Mnemonic / constant

  Example: j 500

1. Decode assembler instructions from the following DLX instructions [3], that they do?
   00000010101011000101000000000110
   01101010010001010000000001101110
2. Code the following DLX assembler instructions as a bit string (DLX instruction):
   xor R3 R5 R12
   subi R1 R10 122

[1] Layout and corrected formulas for the decoder see in the internet page under *Layouts*

[2] Definition of the load and time is in the internet page under *Layouts*

[3] The right most bit is the bit in the position 0

University of the Saarland
Department 6.2 - Informatik
Prof. Dr. W.J. Paul

Computer Architecture I - WS 03/04

(due: 10.12.2003)

Exercise 4: (faster layout) **(12 points)**

This exercise consist of 3 tasks, one task for one exercise group. In each case you need to specify and analyze an alternative layout for one given in the lecture. Your layout should be faster than constructed in the lecture (i.e. minimize a delay).

- **Group 1** (Mo. 16-18)
  Give an alternative layout for the decoder.
- **Group 2** (Wed. 14-16)
  Give an alternative layout for the CLS.
- **Group 3** (Fr. 16-18)
  Give an alternative layout for the Random Logic (see lecture), i.e. to compute t monomials from n input variables.