Prof. Bernd Finkbeiner, Ph.D.
Dipl.-Inf. Rüdiger Ehlers
Markus Rabe, M.Sc.
Sebastian Hahn, B.Sc.

Summer term 2012
Problem Set 8
Due: Wednesday, 20th June 2012

Embedded Systems

Please indicate your **name**, **matriculation number**, **email address**, and which **discussion session** you have been allocated to. We encourage you to collaborate in **groups** of up to **three** students. Only one submission per group is necessary.

Exercise 1: A/D Conversion

In the lecture we discussed the successive approximations method as one possibility for converting analog signals into digital values. Using the successive approximations method, carry out the conversion of the input voltages $U_{in} = 2.8V$, $1.6V$, and $3.55V$ into the corresponding binary values. In each case, give the final binary value, and for each step of the conversion, show:

1. the arranged comparison voltage U_{ref};
2. the binary value for each comparison.

The digital value should have a precision of 4 bits. Assume also that the working range of the A/D converter lies between $U_{min} = 1V(0000_2)$ and $U_{max} = 4V(1111_2)$.

Exercise 2: SIMD

Assume that we have a single-instruction multiple-data (SIMD) processor with a 64-bit input port and a 64-bit output port. The processor does not have any external memory other than the 64-bit registers mm0 to mm7, and the following instructions, where x and y refer to any register, and z can refer to (the value of) any register or any constant:

Command	Explanation
`read x`	Obtains a new 64-bit value from the input port and stores it to register x. Blocks until 64 bits of data are available.
`write x`	Writes x to the output port. Blocks until data can be written.
`jump j`	Jumps to label j in the program
`and x,y,z`	Takes a bit-wise AND of z and the value in register y, and stores it to register x
`or x,y,z`	Takes a bit-wise OR of z and the value in register y, and stores it to register x
`shift x,y,c`	Shifts the content of register y by c (which is a constant between -64 and 64), and stores the result into register x. Fills with 0s.
`paddw x,y,z`	A SIMD instruction: Divides y and z into 16-bit wide words (representing unsigned integer values), adds the ith chunk of y to the ith chunk of z for every $i \in \{1,2,3,4\}$, and stores the results together in register x
`pmullw x,y,z`	A SIMD instruction: Divides y and z into 16-bit wide words (representing unsigned integer values), multiplies the ith chunk of y with the ith chunk of z for every $i \in \{1,2,3,4\}$, and stores the results together in register x

Assume that the processor gets a data stream $\vec{X} = X_0X_1X_2\ldots$ of 16-bit words fed to its input port such that every reading operation reads four values at once. Create a program to let the processor output a stream of 16-bit integers $\vec{Y} = Y_0Y_1Y_2\ldots$ such that for every $i \in \mathbb{N}$, we have $Y_i = (X_i + X_{i+1} + 3) \cdot 2$. You can use labels in your program as targets for the `jump` operation. After reading some values, the output does not have to be produced immediately, but outputting may be delayed. Note that the `write` operation will also output four items of the Y stream at once. You can use binary or hexadecimal notation for all constants if you wish. We assume that for all $i \in \mathbb{N}$, we have $0 \leq X_i < 16000$, and the addition and multiplication operations will behave in some bad non-specified way upon overflows. All arithmetic operations work on non-negative numbers.

Exercise 3: Aperiodic Scheduling

Assume a uniprocessor architecture without preemption. Consider the following set of asynchronous, aperiodic, and independent tasks:

Job	J_1	J_2	J_3	J_4
Arrival time a	0	4	2	6
Computation time C	6	2	4	2
Deadline d	16	10	9	12

Is the given task set schedulable? If it is, your task is to find the schedule.

Exercise 4: FlexRay

Briefly answer the following questions about the FlexRay protocol:

- Why does it make sense to have a static segment and a dynamic segment in a bus protocol?
- What is the difference between glitches and jitter?