Saarland University
Department 6.2 - Computer Science
Prof. Dr. W. J. Paul
Geng Chen

Multicore System Architecture - WS15/16

Exercise Sheet 6 (due: Dec 4, 2015)

Important:

- Each week on Friday one exercise sheet will be released. The solutions should be handed in before or after the next Friday lecture if not stated otherwise. For the admission to the exam you will need at least 50% of the points of the exercises.
- Tutorials: Group A: Wednesday 12:00 - 14:00 (Room 328, E 1 3, Tutor: Jonas Donia) and Group B: Tuesday 10:00 - 12:00 (Room 328, E 1 3, Tutor: Shahd Zahran)
- You are allowed to solve the exercise sheets in groups. Everybody who has his or her name on the solution must be able to present it in the tutorials. Everybody must present the solution of at least two exercises.
- Please, register for the lecture at the lecture's webpage until Nov 13th, 2015!
  http://www-wjp.cs.uni-saarland.de/lehre/vorlesung/rechnerarchitektur/ws15/anmeldung.php
  Also do not forget to register for the exam in the HISPOS system!
- The oral exam will take place in February. An exact date will be decided upon in class.

Tutor: _____

Name, Matr. Number: _____

Exercise 1: **(4)**

We augment master and slave automata as shown in figure x. In state *wt* the master will update the memory after a write. State *sd* is used to switch tri state drivers on bus *bdata*. The corresponding slave states *swt* and *ssd* serve only to keep the automata in sync. Note that master protocol signal *wt* is used here by the slaves to trigger te transition to state *ssd*.

Adjust the definitions of transition (2), (3), (4) and (5). Hints:

- (2) and (3): in write through mode writes are not local.
- (4) and (5): after a write miss in write through mode there is no need to flush, because the line is not cached.

Exercise 2: **(4)**

Specify the control signals generated at the new states. In particular

1. set the output enable bits of drivers to bus *bdata* in state *sd*.
2. update caches states in the last cycle of *wt* resp. *swt*.
3. turn signal *mbusy* off during the last cycle of *wt*.
4. do not forget to disable drivers again.

Exercise 3: **(2)**

Explain, why we do not set enable bits for write back already in the last cycle of *mdata*. Hint: recall sec. 3.5.7.

Exercise 4: **(2)**

In what part of the correctness proof do we use, that we have kept the slave automata in sync with the (redundant) states *ssd* and *swt*?

Saarland University
Department 6.2 - Computer Science
Prof. Dr. W. J. Paul
Geng Chen

Multicore System Architecture - WS15/16

Exercise Sheet 6 (due: Dec 4, 2015)

Exercise 5: **(4)**

In a bus arbiter a circuitry which in every hardware cycle "grants" the bus to exactly one cache. To be granted on the bus, cache i first raises its request signal $req[i]$. Not sooner than in the next cycle, cache i can be granted. In this exercise we implement the round-robin scheduling, of course with respect to the active requests. Consider the arbiter depicted in Figure 5. Give a construction of the circuit which computes the $nextgrant \in \mathbb{B}^{2p}$ signal, s.t.

$$nextgrant[i] \leftrightarrow i = \begin{cases} \min\{j \mid req[j] \wedge j \geq k\} & grant[k] \wedge \exists j \geq k : req[j] \\ \min\{j \mid req[j]\} & \text{otherwise.} \end{cases}$$

Prove correctness of your implementation.