**University of the Saarland**
**Department 6.2 - Informatik**
**Prof. Dr. W.J. Paul**
**Dipl. Inf. Sven Beyer**
**M. Sc. Sergey Tverdyshev**

# Computer Architecture I - WS 04/05

**Cache Accesses**

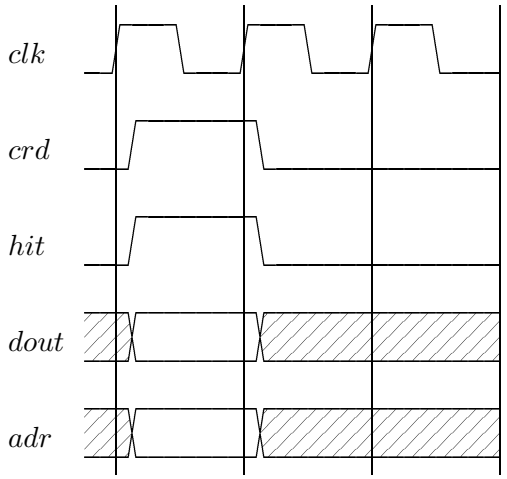


Figure 1: read hit

Figure 1 and figure 2 depict timing diagrams for case read-hit and write-hit respectevily. Figure 3 depicts timing diagram for case read-miss for the data which were not marked as dirty. In this case we do not have to execute write back and directly load data to the cache. During cycles $T1 \ldots T6$ loading data($fill\ line$) is performed. In this example we load data for 2 segments. Afterthat it will be followed by read or write operation. Last picture presents dirty-miss-wrire. In this case the data from the cache have to be written back to the memory and evicted from the cache (cycles $T1 \ldots T5$). Eviction address is given on bus $ev$. After eviction $fill\ line$ is performed.

**University of the Saarland**
**Department 6.2 - Informatik**
**Prof. Dr. W.J. Paul**
**Dipl. Inf. Sven Beyer**
**M. Sc. Sergey Tverdyshev**

# Computer Architecture I - WS 04/05

**Cache Accesses**

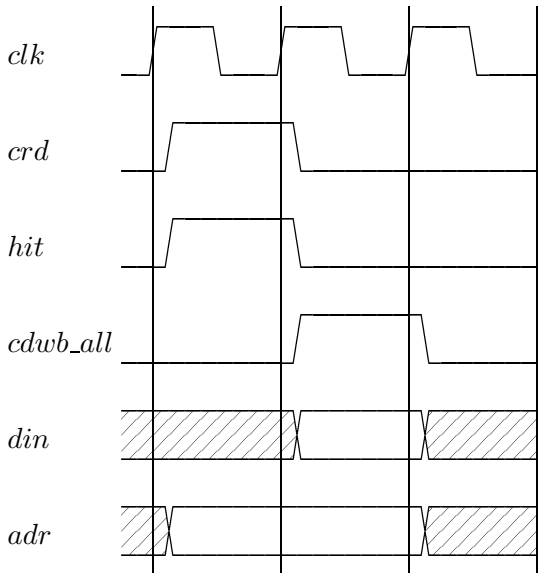


Figure 2: write hit

**University of the Saarland**
**Department 6.2 - Informatik**
**Prof. Dr. W.J. Paul**
**Dipl. Inf. Sven Beyer**
**M. Sc. Sergey Tverdyshev**

# Computer Architecture I - WS 04/05

**Cache Accesses**

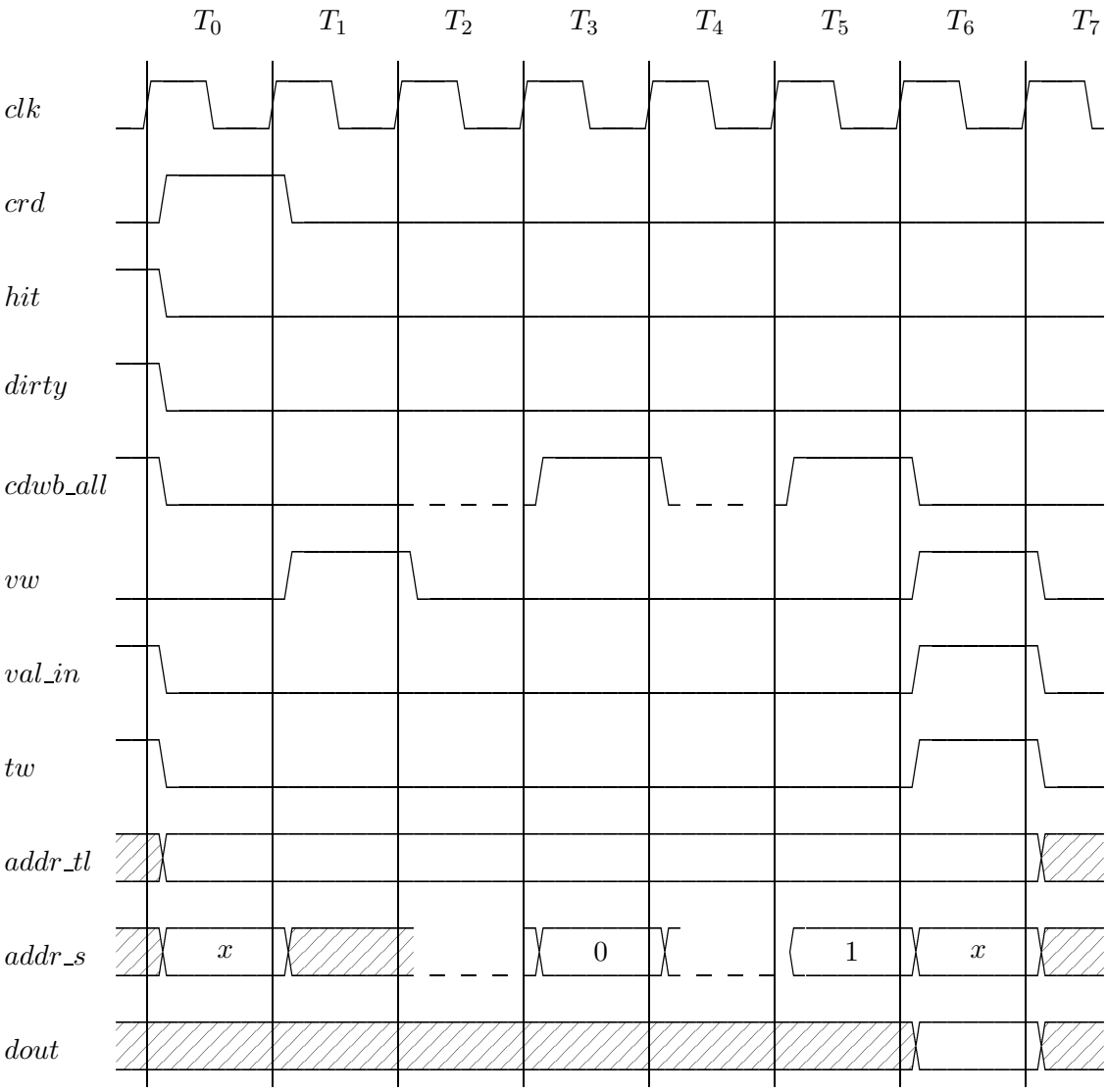


Figure 3: clean miss: read

**University of the Saarland**
**Department 6.2 - Informatik**
**Prof. Dr. W.J. Paul**
**Dipl. Inf. Sven Beyer**
**M. Sc. Sergey Tverdyshev**

# Computer Architecture I - WS 04/05

**Cache Accesses**

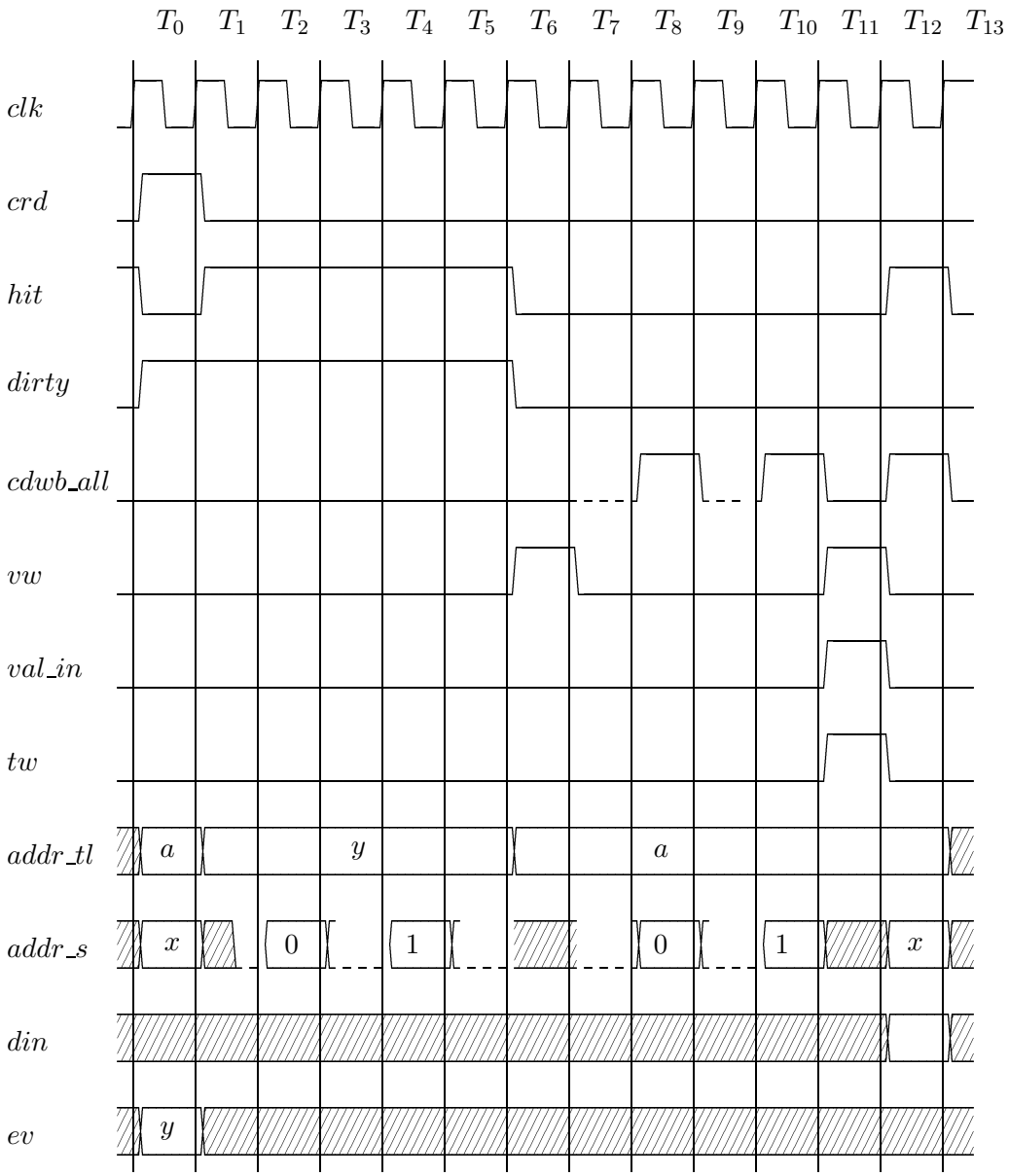


Figure 4: dirty miss: write