DMA

18-1 ECE 322 Computer Organization, Lab 08

Hansung Kim
2014-16824

1 Introduction

Implement a DMA controller on top of the previous CPU implementation. Learn how CPU cooperates with DMA to achieve better performance.

2 Design

The key point of this design is the use of a signal called `d_cache_busy` that indicates whether the cache is occupying the data and address bus, i.e. handling cache miss. The addition of this single signal simplifies the logic of granting of the bus to the DMA.

2.1 Bus grant and reclaim

The CPU can grant the bus access to the DMA whenever the cache is not handling miss, i.e. `d_cache_busy == 0`. Conversely, if the DMA stops requesting for the bus, it is guaranteed that the bus is vacant and the CPU can reclaim the access immediately. Therefore, the code for the bus grant and reclaim logic is simply as follows:

```
if (bus_request && !d_cache_busy) begin
   bus_granted <= 1;
end

if (!bus_request) begin
   bus_granted <= 0;
end
```

Figure 1: Bus grant and reclaim logic using `d_cache_busy`

where `d_cache_busy` is produced by testing if the cache is being read or write but its value is not ready, i.e. `(d_readC OR d_writeC) && !d_readyC`.

2.2 Cycle stealing

The DMA cycle stealing of the CPU can be implemented easily with little modification. After each memory write of 4 words, the DMA sets `bus_request` to zero for exactly one cycle. This causes the CPU to retry any currently blocked memory operation in that single cycle. If there were any, the operation will set `d_cache_busy` on, delaying the `bus_granted` to go up until the operation is finished. If there were no blocked memory operation, the logic in Figure 1 would set `bus_granted` back on immediately. Therefore, the Figure 1 logic is capable of handling cycle stealing by itself.

2.3 Changes to the cache

The cache is modified to not issue any memory read/write command when the bus is granted. Its "ready" state should also remain false for the whole time span of bus grant. Thus, the `readM`, `writeM` and `readyC` signal is additionally logical-ANDed with `!bus_granted`, which effectively no-ops the cache when the bus is granted to the DMA.

2.4 Changes to the hazard unit

The hazard unit stalls the entire pipeline on `d_cache_busy`. This single check remains to be enough for handling both the D-cache miss and DMA bus block, as `d_cache_busy` will be set to zero in both cases. Thus, the hazard unit does not require any modification from the cache implementation (aside from the variable renaming).

3 Implementation

4 Discussion

5 Conclusion