Lab3 Report

Name: Haoyang Zhang

Email: hzhang11@usc.edu

Remote repository address: https://github.com/hzhang2422/EE-533

1. Explain the pattern matching algorithm

The pattern matching algorithm can detect a 7-byte message even when the detected message spans its own beginning and ending boundaries. Additionally, it allows for the free configuration of wildcards to flexibly match the target message.

In each clock cycle, the lower 56 bits (7 bytes) of the 72-bit data stream will be detected. Firstly the data stream is passed through a delay register, and the lower 48 bits (6 bytes) are merged with the previous data and placed in the high bits, which ensures any possible contiguous 56-bit substring is covered. This process is like a slide window searching for target message.

The whole matching pattern is stored in hwregA[63:0]. The lower 56-bit data is target message. Another 7 bits are designated as wildcard bits, each of them corresponding to one of the 7 bytes. Setting a wildcard bit to "0" will cause the corresponding byte to be skipped during matching.

Finally, the matching status is latched to ensure that a stable match signal output once the matching condition is met, until an external reset signal clears the state.

2. Other questions

- **What is the purpose of AMASK[6:0]?**

  AMASK[6:0] is set up as a wildcard to flexibly implement which bytes need to be matched and which to be ignored. The byte would match any value if we set its corresponding AMASK bit to "0".

- **What exactly does busmerge.v do?**

  Busmerge merges the last 6 bytes of previous message and last 7 bytes of current message, which ensure that any consecutive 7-byte data can be detected by slide-detecting window.

- **What do the comp8 modules do in this schematic?**

  COMP8 performs a bit-by-bit comparison of the two inputs, and it only outputs 1 if every corresponding bit is equal.

  In comparator.sch, COMP8 compares the 7-byte data stream with target message.

  In dropfifo.sch, COMP8 compares read address with write address of dual port memory to generate output valid data.

- **What is the purpose of dual9Bmem in dropfifo.sch?**

The dual9Bmem as a dual-port RAM, serving as a data buffer in the dropfifo module. It can be configured to either retain or drop a data packet by controlling the read and write enable signals.

In addition, it ensures that data throughout is carried out effectively at high speed.

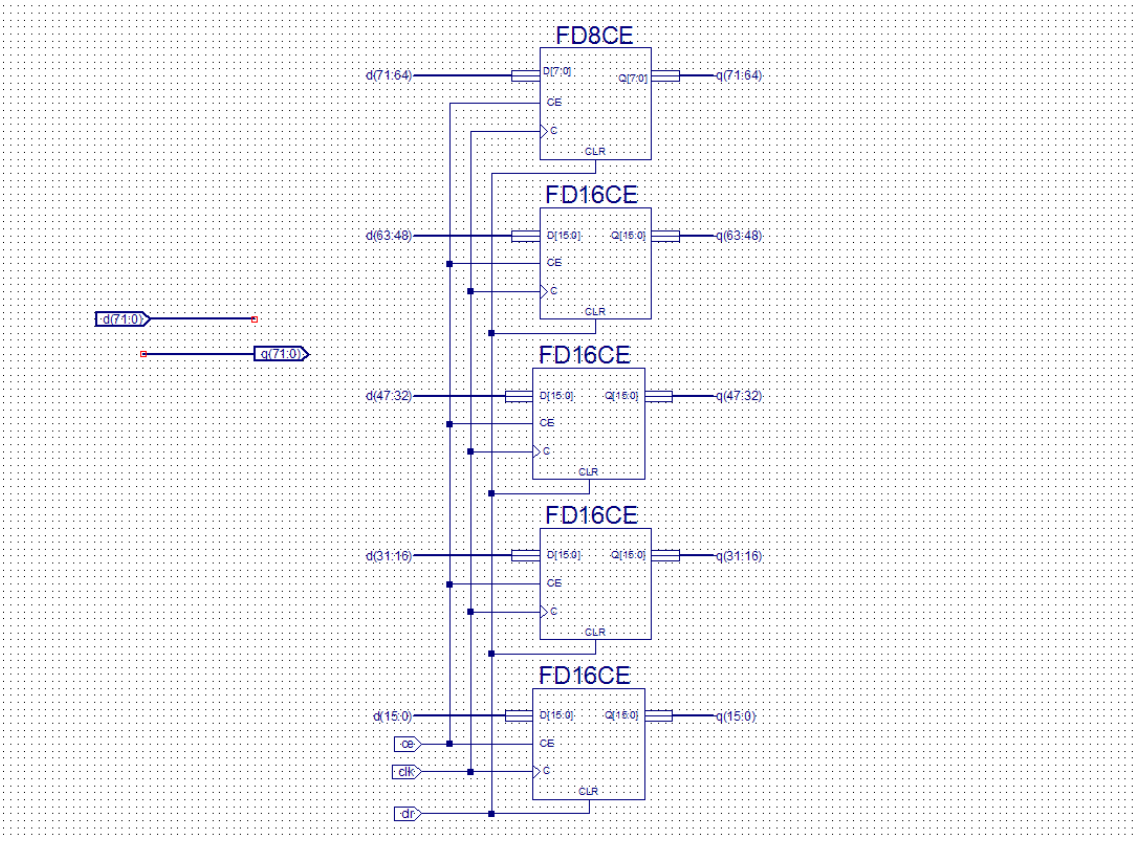
3. The screen capture of Schematics as well as generated Verilog

- busmerge.v

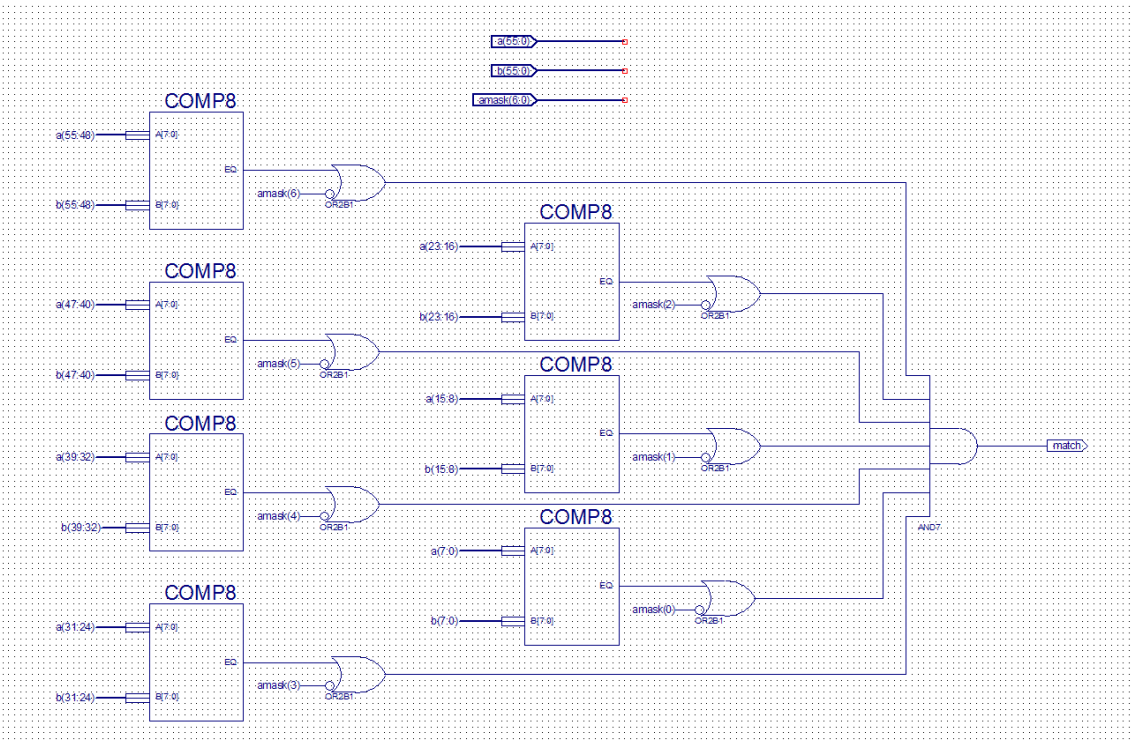
```
//////////////////////////////////////////////////////////////////
module busmerge(da, db, q);
    input [47:0] da;
    input [63:0] db;
    output [111:0] q;

    assign q = {da,db};
endmodule
```

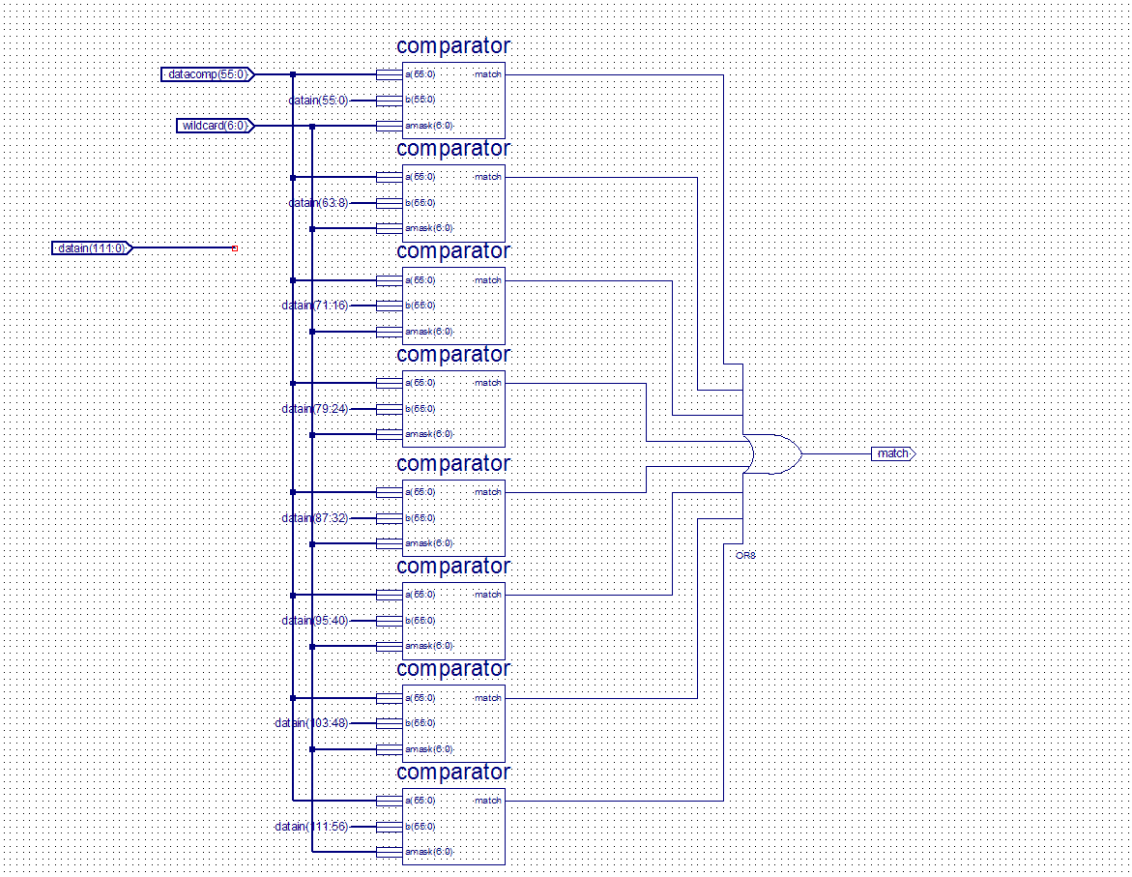
- reg9B.sch



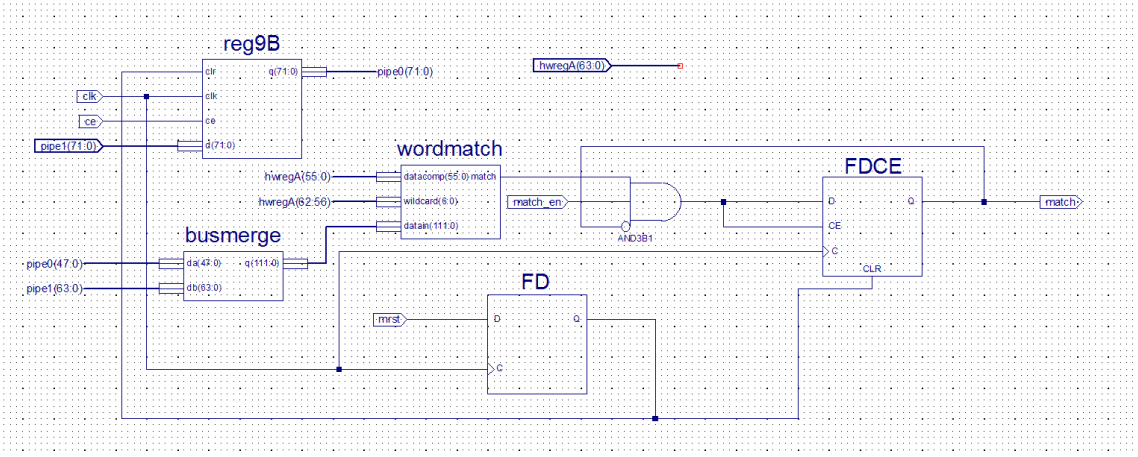
- comparator.sch



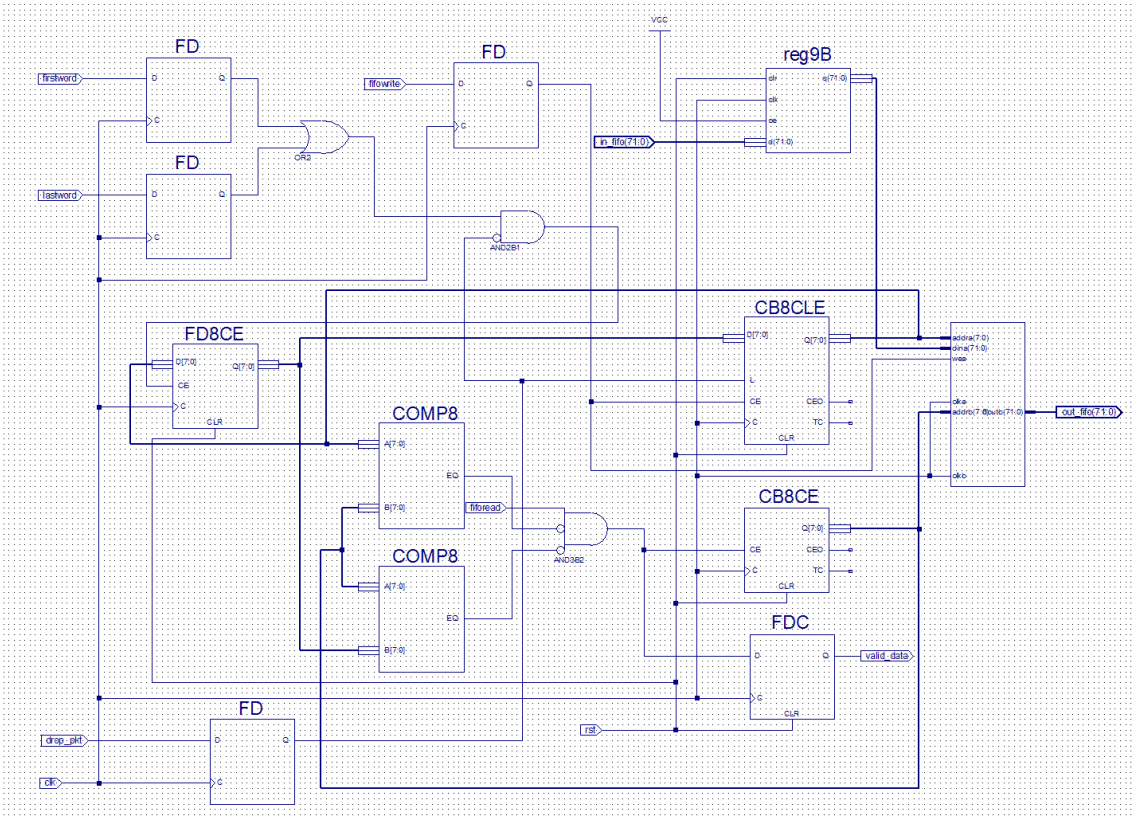
- wordmatch.sch



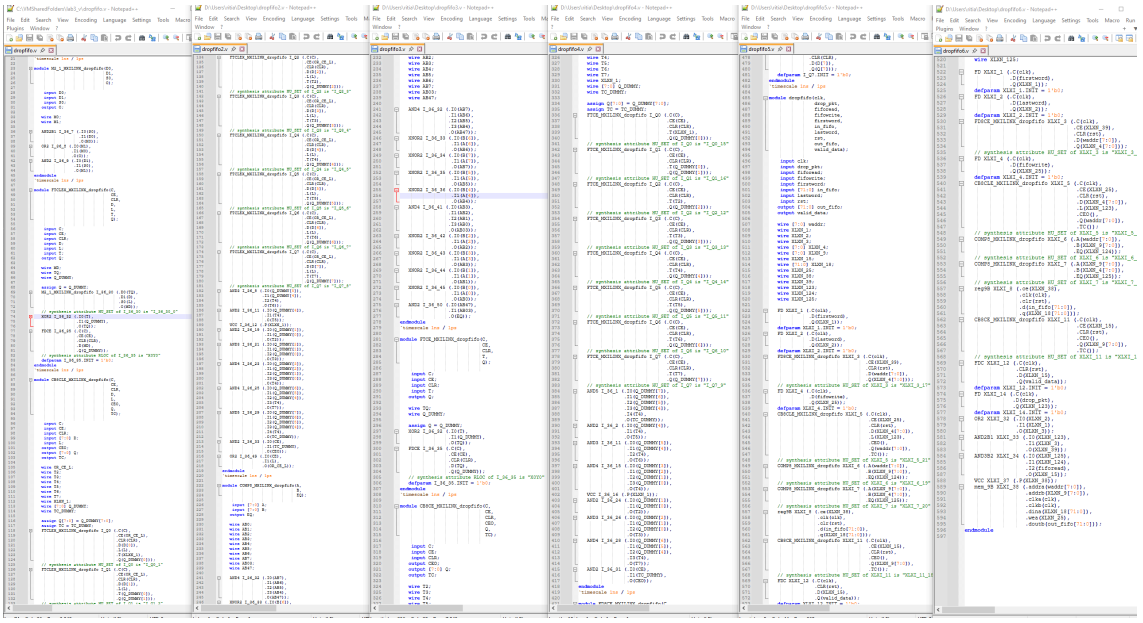
- detect7B



- dropfifo.sch



- dropfifo.v



- Simulation Waveform