بنام خدا

دانشگاه خوارزمی

طراحی سیستم‌های دیجیتال

نیم سال دوم ۱۴۰۱-۱۴۰۲

تمرین سوم

موعد تحویل ۳۱ اردیبهشت ۱۴۰۲

- جواب سوالات را در قالب فایل PDF، به آدرس ”khu.dsd98@gmail.com“ با عنوان "A_G_3: stdName 1, stdName 2"ارسال کنید.
- تمرین ها را می توانید در گروه های دو نفره ارسال کنید که stdName 1 , stdName 2 نام اعضای گروه است.
- **مهم**: فایل PDF باید شامل کدهای VHDL قابل سنتز و اسکرین شات‌های RTL، post-mapping و compilation report فایل Test Bench و اسکرین شات Wave Form از Modelsim باشد.
- برای تکلیف ارسال شده با تاخیر نمره‌ای منظور نخواهد شد.
- تمامی Test Benchهای شبیه سازی باید توسط Modelsim صورت پذیرد.

1. Let’s assume that you are asked to implement a FIFO buffer for a 32-bit system. Write a synthesizable VHDL code describing the FIFO buffer, which includes FIFO controller and register file, for a 4-word FIFO (A FIFO containing 4 slots, each slot is 4 bytes). The FIFO buffer must have 2 components, one register file and one control unit!
    - Then, write a proper Testbench for this FIFO buffer and verify if the FIFO buffer works correctly or not. <u>You need to examine underflow and overflow events in the simulation!</u>

```
entity fifo_Buffer  is
        port (clk, reset: in std_logic;
              wr,rd:instd_logic;
                w_data: in std_logic_vector (32-1 downto 0);
                full, empty: out std_logic;
                 r_data: out std_logic_vector (32-1 downto 0)
                        );
end fifo_Buffer;
```

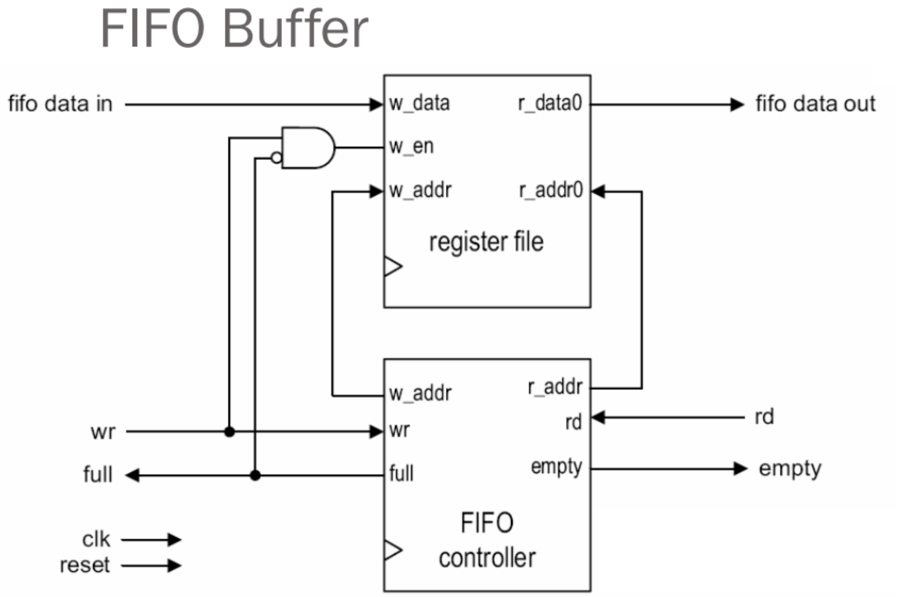


Figure 1:FIFO Buffer

2. Figure 2 shows an FSM which is used by a controller. This question asks you to:

    1- Draw its ASM Chart.
    2- Write a synthesizable VHDL code that implements the FSM.

3- The Moore output must be Glitch-free. To do so, you need to have a look-ahead output buffer.

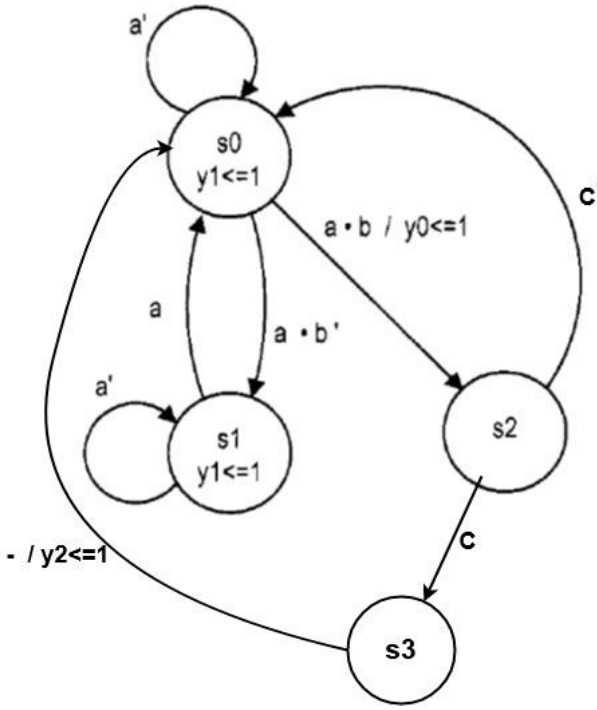
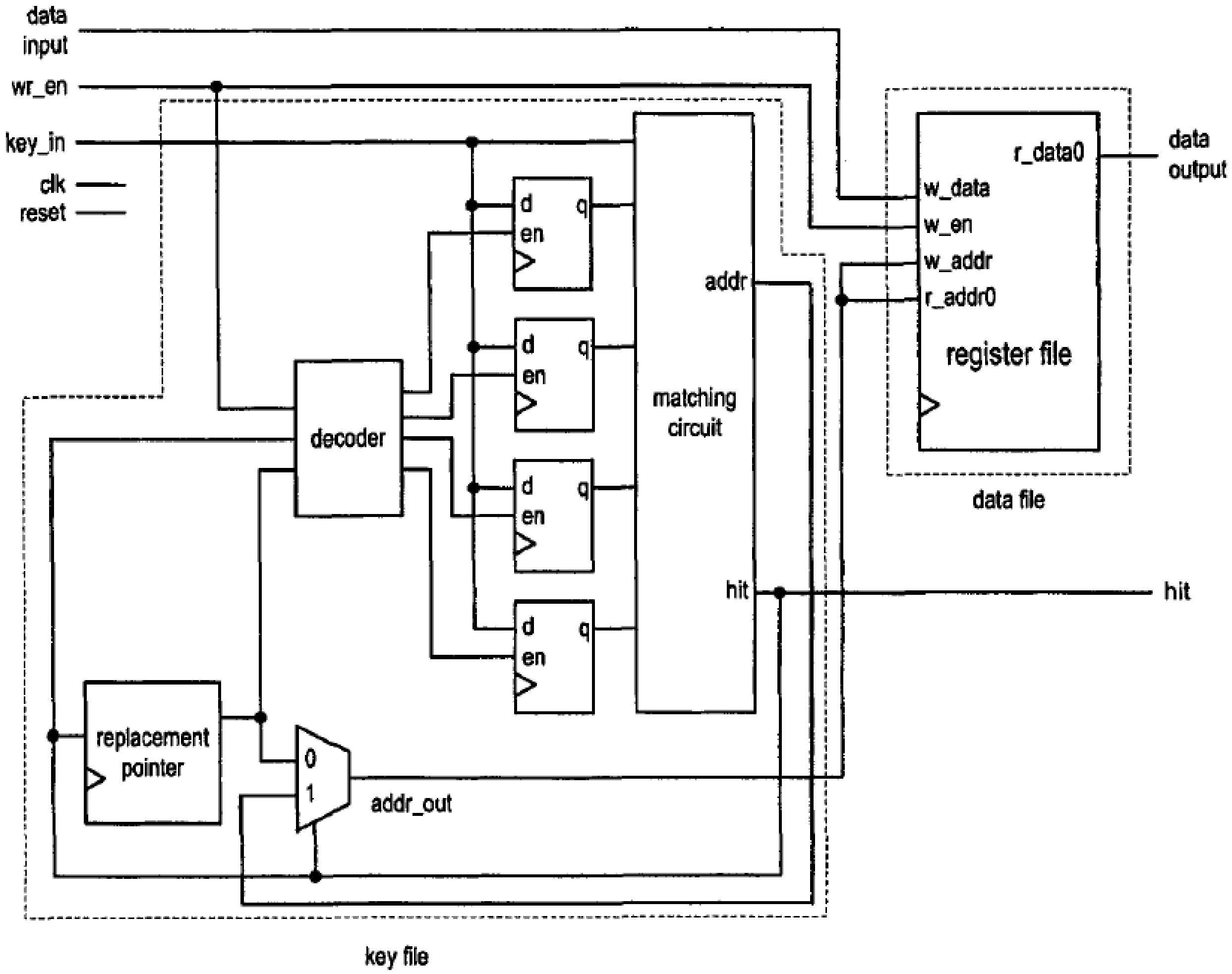


Figure 2: An FSM

3. You need to design a fully qualified test bench to verify the 16-word CAM memory (fully-associative cache), which includes CAM controller and register file, to be used for a 4GB memory. This CAM memory is going to be used in a 64-bit CPU!

```
entity CAM_Mem is
        port (   clk, reset: in std_logic;
                 wr_en      :in std_logic;
                 key_in:  in std_logic_vector (32-1 downto 0);
                 data_in: in std_logic_vector (64-1 downto 0);
                 hit: out std_logic;
                 data_out: out  std_logic_vector (64-1 downto 0)
                 );
end CAM_Mem;
```


data input
wr_en
key_in
clk
reset
d
q
en
decoder
matching circuit
addr
hit
replacement pointer
0
1
addr_out
key file
r_data0
w_data
w_en
w_addr
r_addr0
register file
data output
data file
hit