UMONS
Université de Mons

Control an 8x8 LED Matrix with a DE1-Soc Driver

De Brauwer Alexandre, Kokou Komedja and Yilmaz Murathan

POLYTECH MONS

Hardware/Software Platforms
Mr. Carlos Alberto Valderrama

14 June 2019

Outline of presentation

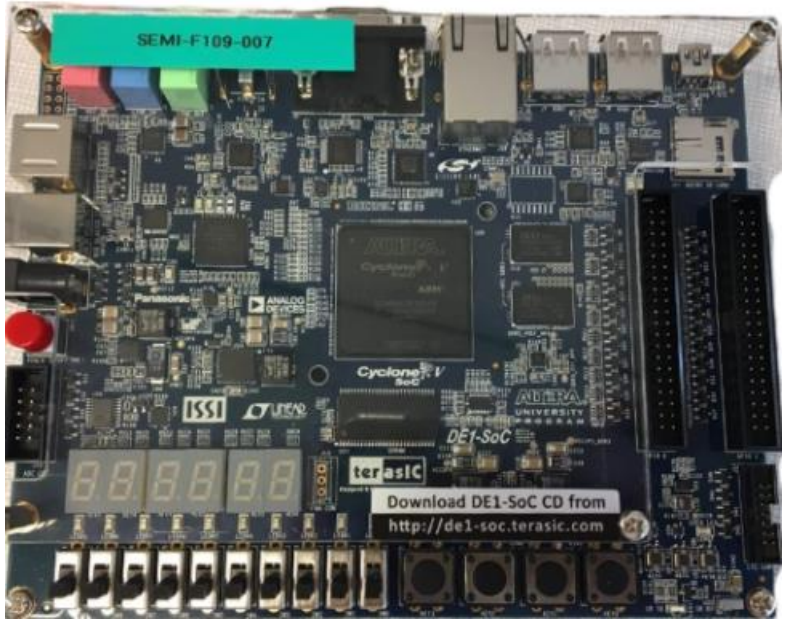
- Introduction
- Hardware
- Software
  - Codes
- Test
- Conclusion

Introduction

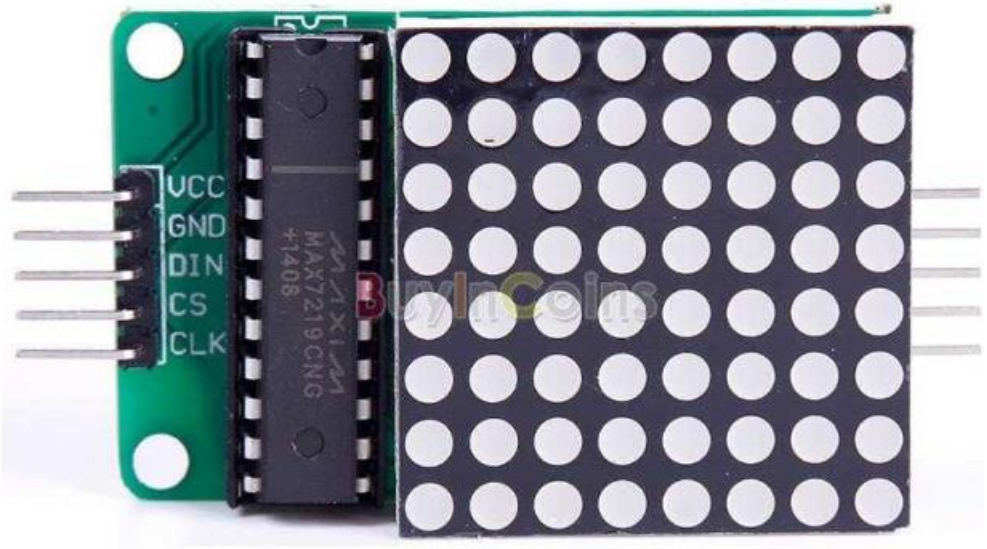
- LED Matrix need to be driven => input of three signal (DIN, CS and CLK)
- FPGA board used as driver => Create these signals and send it to MAX7219
- How can we harmonize all of these together ?

Hardware

- FPGA (DE1-SOC)

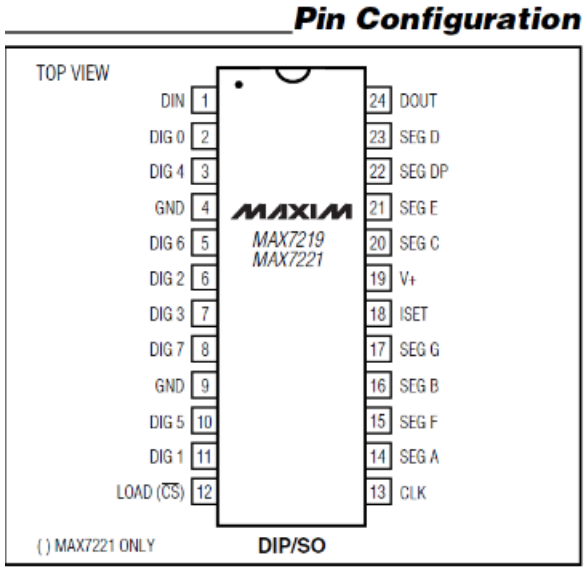


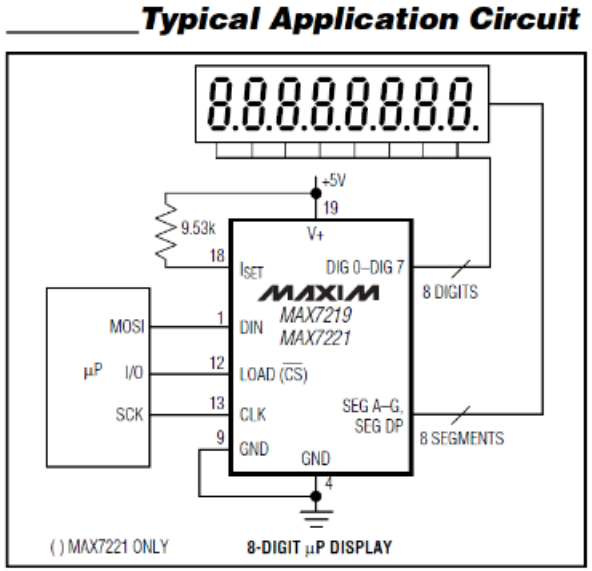
Led Matrix and MAX7219



Software

- Generate the signals (DIN,CS and CLK)=>Pin configuration





Software

- Special format of data is required for the LED Matrix => Data sheet

MAX7219/MAX7221

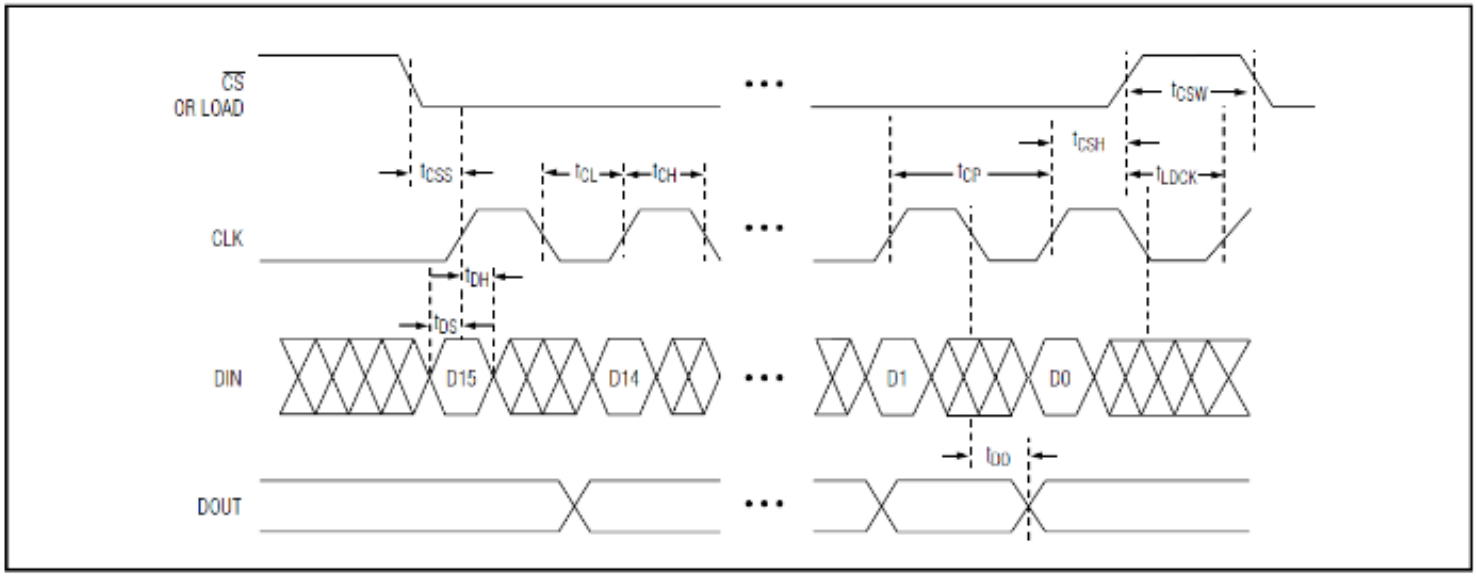


Figure 1. Timing Diagram

Table 1. Serial-Data Format (16 Bits)

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
X	X	X	X	ADDRESS				MSB	DATA						LSB

Software « Codes »

- How our code is done ? => 5 parts

1. Library
   - Fonts => Define the numbers between 0 and 9 in bits for later use in our main code
   - Types => Used to create a state machine with 4 states=> initialize, ready, executeand busy.
   - Virtual clock => Internal clock
2. Spi_master=> Control the channel between the DE1-SOC and the MAX7219.

Software
« Codes »

3. Control of the Max7219 => Generate the signals according to the requirement saw before.
4. Main code =>Update the display and implement the counter (We can display whatever we want).
5. Finally, to make everyting work => Need of mapping between DE1-Soc Pins and Max inputs.

```
    CLOCK_50 => CLOCK_50 ,
    LED_DIN => GPIO_0(1),
    LED_CS => GPIO_0(3),
    LED_CLK =>  GPIO_0(5)
);
```

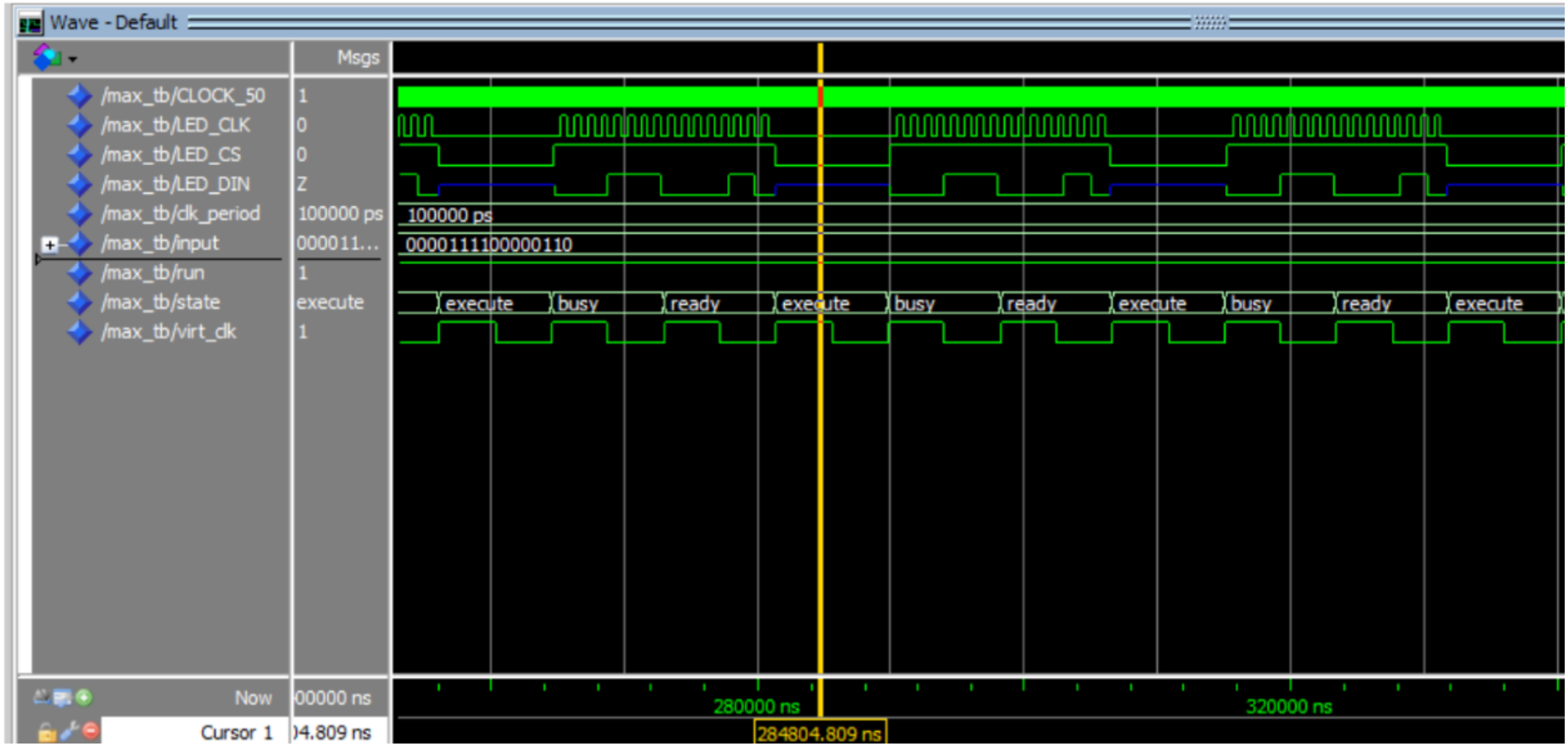
Test

- Creation of a test bench=> Define an input data

```
   SIGNAL CLOCK_50 : STD_LOGIC;
   SIGNAL virt_clk : STD_LOGIC;
   SIGNAL LED_CLK : STD_LOGIC;
   SIGNAL LED_CS : STD_LOGIC;
   SIGNAL LED_DIN : STD_LOGIC;
   signal state   : types;
   signal input           : STD_LOGIC_VECTOR(15 downto 0) := (1 =>'1', 2 => '1', 8 => '1', 9 => '1', 10 => '1', 11 => '1', others => '0');
   signal  run             : STD_LOGIC := '1';
   COMPONENT max is
       PORT (
           CLOCK_50        : IN STD_LOGIC;
           input           : IN STD_LOGIC_VECTOR(15 downto 0);
           run             : IN STD_LOGIC := '0';
           state           : BUFFER types := initialize;
           virt_clk        : IN STD_LOGIC := '0';
           CLK, DIN, CS    : OUT STD_LOGIC
       );
   END COMPONENT;
   constant clk_period : time := 100 ns;
BEGIN
   vclock : entity work.virtual_clock PORT MAP (CLOCK_50 => CLOCK_50, virt_clk => virt_clk);
   i1 : max
   PORT MAP (
-- list connections between master ports and signals
      CLOCK_50 => CLOCK_50,
      input    => input,
      run      => run,
      virt_clk => virt_clk,
      state    => state,
      CLK => LED_CLK,
      DIN => LED_DIN,
      CS =>  LED_CS
   );
   init : PROCESS
   -- variable declarations
   BEGIN
      CLOCK_50 <= '1';
         wait for clk_period/2;
      CLOCK_50 <= '0';
         wait for clk_period/2; -- code executes for every event on sensitivity list
   END PROCESS init;
END max arch;
```

Test

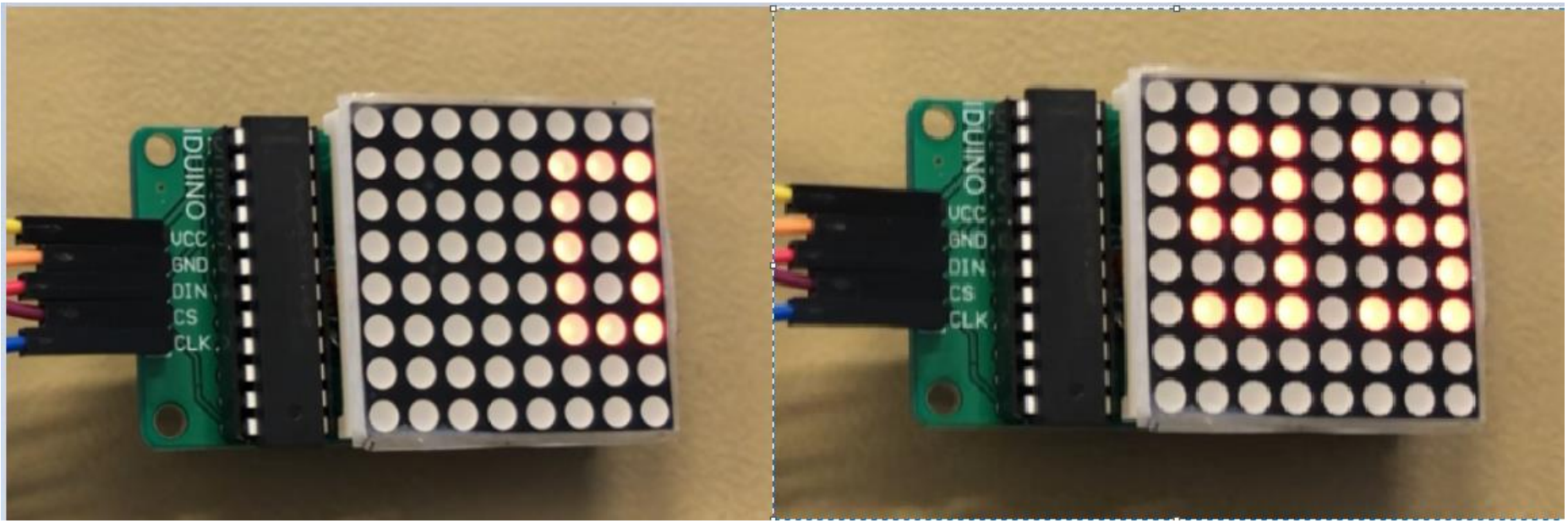
- Simulation results



Test

Flashing the code on the board and

VOILA !



Conclusion

- As we saw, controlling the LED Matrix is not the difficult part
- The most important part is how to link the things together=> Control the channel between MAX and DE1-SOC
- Once all the settings are done, just need to change the main code to display what we want (Letter, number, counter,…)

Thank you for your attention !

Questions ?