CSD example problems and projects

Chapter 2: sequential systems

This is the former CSD problems collection. It is not currently updated, kept only as a reference for old links.

The idea is to transfer and rewrite problems as projects to be organised and saved as CSD web pages.
2.1. Water tank controller

a. Study the specifications. We want to design a water tank controller \textit{(Water\_tank\_controller)} as an FSM that can drive two pumps independently, as represented in Fig. 1. The tank has level sensors D1, D2, and D3 attached to the wall, so that a ‘1’ is generated when the sensor is sunk into water. The controller works as follows: when it is empty, below D1, both pumps work simultaneously; when the water level is above D2 pump P1 stops; when the water is above D3, meaning that the tank is full, pump P2 stops; and finally, the pumps do not switch on until the water level is again below D1.

![Diagram of the water tank installation.](image)

This link is an example of a state diagram.

b. Plan: FSM. Draw the state diagram if, in addition to controlling the water level, we also want to indicate in a LED column the current level of the water in the tank.

c. Plan: adapt the general FSM architecture to this problem and draw the state register based on D\_FF. Deduce how many D\_FF are required if you are coding in binary sequential or in one-hot.

d. Plan: write the truth table of CC1 and CC2 and their equivalent behavioural interpretations in flowcharts.

e. Development: write the VHDL file \textit{Water\_tank\_controller.vhd} by translating the flowcharts and the state register. Run a project using an EDA synthesis tool for a CPLD or FPGA target chip. Print and discuss the RTL and the technology schematics. The CLK oscillator is 1 MHz.

f. Test: simulate the circuit using a VHDL test bench and discuss the results. Measure the maximum CLK frequency that can be applied to your design considering a target chip from Lattice Semiconductor (ispMach4128V
TQFP100), Intel (Cyclone IV EP4CE115F29C7), or Xilinx (Spartan-3E XC3S500E-FG320).

**g. Additional features added to the basic prototype.** The user wanted to add an extra circuit to translate the LED column code into a 7-segment display. Thus, an additional combinational circuit CC3 is required to meet this new specification. Let us solve the problem using the ROM method for implementing logic functions. The wiring in Fig. 2 shows the naming conventions for the vector HEX0(6..0) common anode in the DE2-115 board user guide page 36. Discuss the size [2^m x n] of the ROM, its content and synthesise the circuit.
2.2. Stepper motor controller

Reinvent project on steeper motor sequence using VHDL techniques as a canonical FSM.
2.3. 7-segment digit sequencer

We want to design a simple driver to shown a sequence of movement, clockwise and counter-clockwise, in a single 7-segment display. Fig. 3 represents the schematic diagram of the application. The circuit components are: (1) a clock to produce a rectangular wave with a given frequency, let us take for instance 5 Hz; (2) the digital system named **sequencer** and (3) the 7-segment display (common cathode) with its current-limiting resistors.

The system has to work as specified in Fig. 3c, depending on the logic levels of the synchronous input signals: UD_L (Up – active high / Down – active low) and ST (start/stop). A start pulse (ST) activates the sequence of LED lighting that never ends until another pulse ST is applied and the sequence reached the last state. Because of the requirement that the sequence must end (for example when going UP reaching the state Blank) before stopping (going Idle) if another ST pulse is detected, the design must include a 1-bit memory cell such as an **RS_Latch** or an **RS_FF** and the FSM that generate the sequence and controls the system. Therefore, this will be a plan C2 system composed of a top design (**sequencer**) and some components.
1. Deduce a circuit for solving this problem. This is an initial discussion.
2. Particularise the internal FSM component architecture to this problem, naming and connecting all the inputs and outputs. How many D_FF of memory are used in this problem if coding the state machine in one-hot?
3. Infer and draw the circuit’s state diagram. Annotate all the state transitions and outputs.
4. Sketch a timing diagram showing the main operations. In addition to the ports, include as well internal signals like STB in the discussion.
5. Draw the state register if coding the machine in binary sequential.
6. Write the CC2 truth table to obtain the outputs of the circuit and its flow chart.
7. Design the CC1 truth table to obtain the next state to go and its flow chart.

------------------------- Development and test -------------------------
8. Write the VHDL files (this is a plan C2 design) and run the EDA project to synthesise the circuit and obtain results. Inspect the RTL and verify that it looks like your schematic. Check the number of D_FF, print and comment the schematics.
9. Write a VHDL test bench and run the EDA simulation tool to verify your design.
10. The target chip is the ispMACH4128 which has DFF with a $t_{CO} = 2.7$ ns and logic gates with a $t_{PD} = 2.7$ ns. Which may be a good estimation of the maximum frequency of operation? Explain your answer.

Extra (P8 content on CLK generators: counters and frequency dividers using the plan Y)
11. Design a circuit to produce the 5 Hz square wave from a 50 MHz quartz crystal oscillator and deduce the number of D_FF that will contain.
2.4. Synchronous universal 4-bit binary counter

Our goal is to design as a very versatile building block that can be used in other designs as a component. It is a synchronous, presettable (LD, parallel load), 4-bit (modulo 16), reversible (up and down), binary counter with count enable (CE) and terminal count (TC16) as represented in Fig. 4. It is a chip similar to the classic 74LS169.

![Fig. 4 Synchronous 4-bit universal binary counter. Symbol and function table. This is an example Proteus simulation of a very similar circuit.]

<table>
<thead>
<tr>
<th>LD</th>
<th>CE</th>
<th>UD_L</th>
<th>Q*</th>
<th>Synchronous operation after the CLK's rising edge</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>x</td>
<td>x</td>
<td>Din</td>
<td>Parallel load (register data)</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>x</td>
<td>Q</td>
<td>Do nothing (inhibit)</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>(Q+1)_{mod16}</td>
<td>Up counting in binary</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>(Q-1)_{mod16}</td>
<td>Down counting in binary</td>
</tr>
</tbody>
</table>

TC16 = ‘1’ when CE = ‘1’ and \((Q = 15 \text{ and } UD_L = ‘1’) \text{ or } (Q = 0 \text{ and } UD_L = ‘0’)\); ‘0’ otherwise

Specifications
1. Symbol, function table, example of a timing diagram, state diagram. How many states this sequential system must have?

Planning (plan Y)
1. Customise the general FSM architecture for this problem indicating where every input and output is connected. Plan the circuit as a FSM in a single VHDL file.
2. How many data flip-flops (D_FF) are required? Draw the schematic of the state register. Which is the internal encoding the current_state signal?
3. Name the circuit Counter_mod16 and plan the circuit as a FSM in a single VHDL file.
4. Write the truth table for the CC2 and propose an internal circuit.
5. Write the truth table for the CC1 and propose an internal circuit. Draw the truth table’s flow chart as the behavioural interpretation.

Developing
6. Translate the circuit schematic to VHDL and start an EDA project to synthesise it for a given CPLD/FPGA target chip.
7. Inspect and analyse the RTL view and technology schematics. Check the number of DFF registers.

Test
8. Run a functional simulation translating the timing diagram into a VHDL test bench.
9. Run a gate-level simulation and determine propagation time from CLK to output (t_{CO}) and thus, the maximum frequency of operation for the given target chip.
Problem discussion.
This picture is an example of a functional simulation of the universal counter modulo 16. The comments in red ink are very important to check whether the circuit works as expected.

2.5. Synchronous modulo 12 counter
Design as a complete project following the usual steps, the synchronous up/down modulo 12 counter represented in Fig. 5 using 2 different strategies. Compare and discuss the advantages and drawbacks of each strategy.

![Fig. 5 Synchronous up/down binary counter modulo 12 with asynchronous clear direct.](image)

Project X. Single VHDL file (flat) FSM based on naming states and the use of State_type enumerated signals. Thus, using this methodology this project becomes simply another P6 FSM example.

Project Y. Single VHDL file (flat) FSM based on the arithmetic VHDL library and the use of std_logic_vector signals. Thus, using this methodology this project becomes another exercise like the Counter_mod16 in 1.2.4.

Project C2. Hierarchical structure (multiple file project) based on the building block Counter_mod16 engineered in problem 1.2.4.

Problem discussion and project files.

2.6. Hour counter for a real-time clock
Our goal is to design an hour counter to be used in a real-time clock device to count the hours in modes 0 – 12 (M = ‘1’) and 0 -24 (M = ‘0’). The Fig. 6 represents the schematic diagram of the application when connected to 7-segment digits.
1. Explain the function table of the `hour_counter` discussing the different modes of operation.
2. Draw an example of timing diagram. How many states will the hour counter contain?
3. Draw the function table and symbol of a synchronous 4-bit binary universal counter (`Counter_mod16`).

Plan
4. Organize the internal architecture of the hour counter based on the use of universal 4-bit binary counters (`Counter_mod16`) and combinational circuits and logic gates.
5. How many VHDL files will be required to complete the `Hour_counter_top` in Fig. 6?

Develop
6. Find the `Counter_mod16.vhd` file and translate the hierarchical schematic in Fig. 6 to VHDL.
7. Start a synthesis project and inspect the RTL and technology views schematics. Check the project summary to verify the number of DFF. The target chip may be any CPLD or FPGA available in the laboratory.

Test
8. Translate the timing diagram in 2) to VHDL (`Counter_mod16_tb.vhd`) and run the functional test.
9. Run a gate level simulation and measure the $t_{CO}$ parameter and thus, the maximum speed of operation.

Prototype
10. Choose a laboratory experimentation board like the NEXYS 2 from Digilent. Assign pins and build and check the prototype `Hour_counter_top` adding the necessary chips and modifications. Pay attention on how the 7-segment digits are wired.

+ Problem discussion.
This picture is an example of a functional simulation of the *Hour_counter* working in the AM-PM mode (M = ‘1’).

### 2.7. 6-bit binary universal counter

**Specifications**

Our aim is to design the block represented in Fig. 7, a 6-bit synchronous binary counter (*Counter_mod64*) fully equipped with many features to make it versatile as a component in many projects.

![Fig. 7. The sequential block to be designed and its function table.](image)

<table>
<thead>
<tr>
<th>LD</th>
<th>RST</th>
<th>CE</th>
<th>UD_L</th>
<th>(Q^*)</th>
<th>Synchronous operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>Din</td>
<td>Parallel load (register data)</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>x</td>
<td>x</td>
<td>0</td>
<td>Reset</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>x</td>
<td>Q</td>
<td>Do nothing (inhibit)</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>((Q+1)_{mod64})</td>
<td>Up counting in binary</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>((Q-1)_{mod64})</td>
<td>Down counting in binary</td>
</tr>
</tbody>
</table>

\(TC64 = '1'\) when \(CE = '1'\) \(and ((Q = 63 \text{ and } UD\_L = '1') \text{ or } (Q = 0 \text{ and } UD\_L = '0'))\); ‘0’ otherwise.

1. Draw state diagrams for the circuit to describe the counter’s different modes of operation. How many states will this FSM contain?
2. Draw a timing diagram to represent the different modes of operation: Load, RST, count UP, count DOWN and do nothing.

Planning as a FSM in a single VHDL file (plan Y)
3. Draw the architecture of the FSM particularised for this problem and indicate where all the inputs and outputs are connected.
4. How many registers (D-type flip-flops) will the system include? Explain your answer.
5. Draw the truth table and flow chart of the combinational circuit CC1
6. Draw the truth table and flow chart of the combinational circuit CC2.

**Development using CAD/EDA tools**
7. What is the synthesis process? In which way can we examine the result of the synthesis process?

**Verification:**
8. Write the main features of a test bench file “Counter_mod64_tb.vhd” necessary to perform a functional simulation. For instance, translating to VHDL some vectors from the 2 section.
9. What are the “VHO” and the “SDF” files? Which is the use of these files? The target chip is the ispMACH4128V which has D_FF with a $t_{co} = 2.7$ ns and logic gates with a $t_{ro} = 2.7$ ns. Which may be a good estimation of the maximum frequency if operation? Explain your answer.
2.8. Johnson counter
Design a synchronous 5-bit Johnson counter with count enable and reversibility control signals as shown in the symbol in Fig. 8 using both the Plan X (FSM strategy enumerating/labelling states) and the Plan C2 (using an standard component like the Counter_mod16 in problem 1.2.4).

![Johnson Counter Mod16 Symbol](image)

**a)** Draw the Johnson code for 5 bits. Draw the state diagram indicating both, transitions and outputs. Draw the function table for this counter. Draw an example of a timing diagram.

**Plan X:**

**b)** Draw the FSM structure consisting of CC1 and CC2 and the state register. Indicate its inputs and outputs. How many D_FF will contain the state register if the internal states are coded in binary, and if they are coded in one-hot?

**c)** Draw the CC1 truth table and its equivalent behavioural representation in a flow chart.

**d)** Write the main VHDL sentences of CC2.

**Plan C2:**

**e)** Propose an internal architecture based on the Counter_mod16 and other blocks combinational blocks. A good idea is to design firstly the counter with only up counting direction. And secondly, complete de counter adding the reversibility feature.

**f)** Annotate completely the schematic, chips, signals, etc. and leave it ready for VHDL translation. How many VHDL files will this project include? How many D_FF will this counter include?

For both design plans:

**g)** Write the VHDL code copying and adapting a similar example from the digsys web. Run the EDA synthesis to inspect the
RTL and technology views. Print them both and annotate comments. Check the number of D_FF used.

h) Use a VHDL test bench and run an EDA functional simulation to check how the circuit behaves in time. Print the timing diagram and make annotations.

i) Which is the maximum frequency that can be assigned to the CLK signal when performing a functional simulation?

j) Which is the maximum frequency that can be assigned to the CLK signal when performing a gate-level simulation if the target chip is an Altera CPLD Max II EPM2210F324C3 that has the following characteristics:

<table>
<thead>
<tr>
<th>MAX II Device Features</th>
<th>EPM2210</th>
</tr>
</thead>
<tbody>
<tr>
<td>$t_{PD}$ (ns)</td>
<td>1.7</td>
</tr>
<tr>
<td>$t_{CO}$ (ns)</td>
<td>4.6</td>
</tr>
</tbody>
</table>

Problem discussion using plan X.
2.9. PIC18F4520 TMR2 prescaler design

Design (specify and plan) the programmable Post scaler available in a PIC18F TMR2 represented in Fig. 9 using VHDL techniques and structural plan C2 for a target FPGA chip. Use standard sequential and combinational circuits and logic gates. Let us rename the symbol and ports: TMR2 is the CLK input to the circuit, T2OUTPS(3..0) is the frequency division selector Div(3..0), and Set TMR2IF output is the terminal count TC.

The Post scaler can divide from 1:1 to 1:16 depending on Div(3..0) binary value from “0000” to “1111”. For instance, if FCLK = 15 kHz and Div(3..0) = “0100”, the circuit becomes a 1:5 frequency divider generating FTC = 3 kHz; if FCLK = 84 MHz and Div(3..0) = “1011”, the circuit becomes a 1:12 frequency divider generating FTC = 7 MHz.
2.10. Generation of CLK signals

The sketch in Fig. 10 represents the internal architecture of the *CLK_generator* block that was built to obtain the required CLK signals for a traffic light controller FSM when connected to the UP2 board quartz crystal oscillator of 25.175 MHz.

a) How many DFF registers will require the circuit?
b) How many VHDL files will require the implementation of the circuit?
c) How can you speed up the simulation of the circuit using an EDA tool?

d) Explain how to plan and design a similar circuit to obtain the squared signals of 2.5 kHz (CLK_2_5kHz_SQ) and 10 Hz (CLK_10Hz_SQ) from an *OSC_CLK_in* of 24 MHz.

e) Explain how to design a circuit like the Chip 2 FREQ_DIV_25 in Fig. 10 using VHDL and the FSM technique (plan Y). Represent a timing diagram to show how it works.

f) Write the VHDL code for the circuit and implement the system using EDA tools.
2.11. Pulse generator

In Fig. 11 there is the symbol of a synchronous sequential machine to generate a burst of digital pulses. The proposed circuit is an adaptation to CSD of the original idea from this book Error! Reference source not found.. The timing diagram in Fig. 12 represents how a number of pulses (0, 1, 2 or 3) are generated after triggering the Start_PB input. It can also be seen how an end of operation flag (EO_Flag) is issued to indicate that the machine is no longer occupied and can be triggered again.

![Timing Diagram](image)

<table>
<thead>
<tr>
<th>Sel_pulses_SW(1..0)</th>
<th>Number of pulses</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>0</td>
</tr>
<tr>
<td>01</td>
<td>1</td>
</tr>
<tr>
<td>10</td>
<td>2</td>
</tr>
<tr>
<td>11</td>
<td>3</td>
</tr>
</tbody>
</table>

The architecture in Fig. 13 fulfil the specifications; it is based on a FSM (Chip1) and other components. Why a component such a synchronous data register (Chip2) is necessary? Why a circuit like the debouncing filter (Chip4) is required to interface the Start push-button?

Explain the internal architecture of the Data_register_2bit component. How many states does it have? How many data flip-flops (DFF) are required to implement its state register?

Infer the state diagram of the Pulse_Gen_FSM that may solve the sequence of operations to generate the outputs.

Draw the CC2 truth table and its flowchart interpretation, so that it can be coded in VHDL in the usual CSD style.
Draw the CC1 truth table and its flowchart interpretation, so that it can be coded in VHDL in the usual CSD style.
Draw the internal circuit of the state register. How many data flip-flops (DFF) are required to implement it if we encode the machine in binary (sequential), or alternatively in one-hot?
Write down the VHDL test bench translating approximately the inputs signals in the Fig. 12 diagram.
If the FPGA used as a target chip to synthesise the system has a worst-case time to output propagation delay ($t_{CO}$) of 5.6 ns, which is the maximum CLK frequency and so the minimum pulse duration?
If the circuit uses a 16 MHz $OSC_{CLK\_in}$ from a quartz crystal, invent a CLK generator block to produce both square signals, a system CLK of 10 kHz and a 100 Hz signal to drive the debouncing circuit. How many VHDL files may it contain?
### 2.12. Designing an industrial application

In a gym and fitness centre, there are some shower stalls like the one represented in Fig. 14 that have to be automated to generate cycles of contrasting hot (48 °C), warm (26 °C) and cold (4 °C) water sprays simply clicking a single start push button (SB). After clicking the SB, initially warm water flows for 50 s (H = C = ‘1’), then hot water (H = ‘1’, C = ‘0’) for 10 s, and thirdly cold water (H = ‘0’, C = ‘1’) for 20 s, and this cycle is repeated another time; finally, the system goes idle (H = C = ‘0’) to wait for another user service. During the operation the R_LED turns on and the water solenoid valve (SV) is on. Let us design the digital control system connected to the valves’ power driver (Chip5) using two technologies: a CPLD/FPGA and a microcontroller.

---

![Photograph and sketch of the shower installation to be automated.](image)

---

**Fig. 14**

Photograph and sketch of the shower installation to be automated.
1. Explain the function of the circuit Chip1, why it is required for conditioning the start button external signal.

With respect to the Chip4:
2. Draw the state diagram of the application explaining both, state transitions and outputs in each state.
3. Draw an example of a timing diagram.
4. Draw the architecture of a FSM for the Chip4 explaining where all the inputs and outputs are connected.
5. Draw the CC2 truth table and its flowchart interpretation, so that it can be coded in VHDL in the usual CSD style.
6. Draw the CC1 truth table and its flowchart interpretation, so that it can be coded in VHDL in the usual CSD style.
7. Draw the internal circuit of the state register. How many bits and DFF (data-type flip-flops) will be used if the states are coded in one-hot?

With respect to the Chip3:
8. Explain the internal architecture, components and the number of VHDL files of the Prog_Timer project.

With respect to the Chip2:
9. Deduce and explain the internal architecture, the number of VHDL files and names of the CLK_Generator project.
10. We have measured by means of a gate-level VHDL simulation for the target CPLD/FPGA where the circuit is synthesised, a worst-case CLK to output propagation delay ($t_{CO}$) of 6.3 ns. Thus, which is the maximum $OSC_{CLK\_in}$ frequency and so the minimum pulse duration?
2.13. **Design a 2-digit even/odd counter with start/stop button**

The idea is to design a 2-digit BCD counter that counts even or odd numbers. Fig. 15 shows the main ideas of the specifications. Using the same ST button for start and stop operations makes it possible to plan the systems as an advanced circuit based on datapath and control unit to better handle counting operations.

These is a feasible plan that relies on previous CSD components and projects. You have two options to design the binary Counter_mod50:

a. Using the plan Y in a single VHDL file.

b. Using a plan C2 and components Counter_mod16 and other logic.
2.14. Steeping motor control based on a dedicated processor
This is the project to review and write as a problem: (ref.)