Exam 2 **January 8, 2019**

Problem 1 (Ch2). Analysis.

We aim to design a digital delay generator, a device very similar to a timer, but which can be enhanced in many ways to become a professional instrument priced up to \$4,400 and used in many telecommunications and applied physics laboratories. See Fig. 1. Once triggered (**Start**), a single clock **Pulse** output is produced after a programmable (**Max_Count + 3**) number of **CLK** clock periods (T_{CLK}).

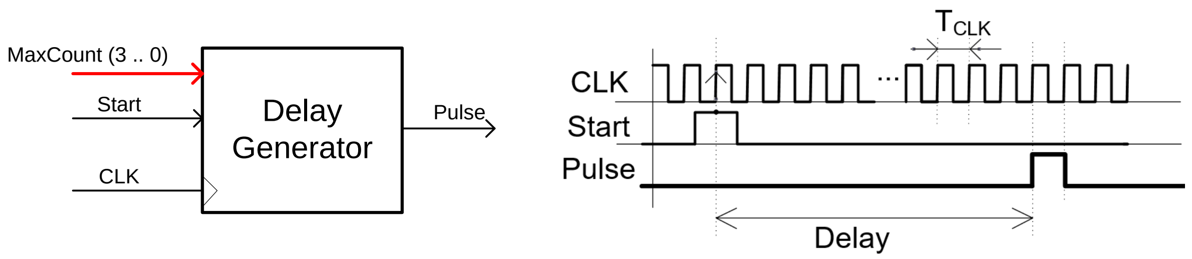


Fig. 1
Digital delay generator symbol and representative waveforms.

The planning of this device is based around a dedicated processor architecture as represented in Fig. 2, which consist of a datapath (ALU + data registers + multiplexer) and a control unit (FSM). Fig. 3 shows the state diagram executed by the FSM in order to perform the programmable delay with such hardware resources. Some components like the Quad_MUX2 and the ALU can be designed following guidelines from Chapter I and other components like the FSM and the data registers can be implemented as learned in Chapter II.

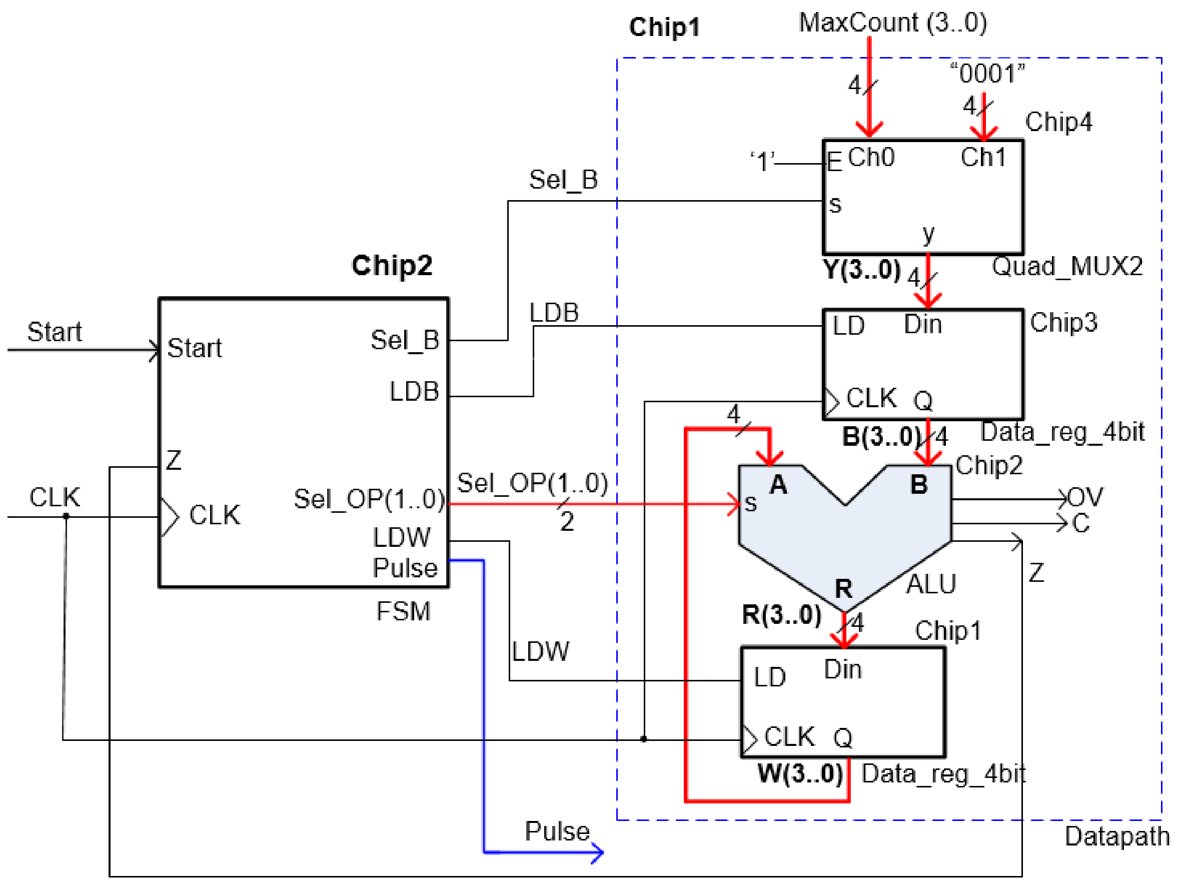


Fig. 2
Proposed architecture for the digital delay generator.

The CD signal is connected to the FSM and the data registers in the datapath.

a) Which is the number of source VHDL files involved in the project assuming the hierarchical plan depicted Fig. 2Fig. 3? Justify your answer.

b) Once synthesised as a flattened gate-level circuit, which is the number of D_FF registers used in this project, if "one-hot" is chosen to encode the state machine? Justify your answer.

c) Draw, paying attention to the state transitions, the timing diagram using an sketch like the one depicted in Fig. 5 whilst adding meaningful comments to the waveforms. Assume that T_{CLK} = 1 µs.

d) Draw the internal architecture of the control unit (a FSM) and explain where all the inputs and outputs are connected.

e) Draw the flow chart for the process CC2 which has the truth table represented in Fig. 3. Write some VHDL statements for this process.

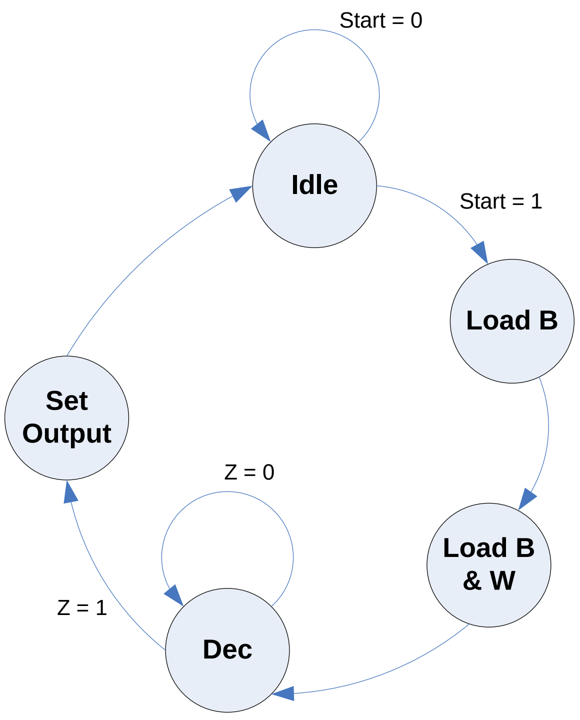
f) Draw the truth table and the corresponding flow chart for the process CC1. Write some VHDL statements for this process.

g) Write down the VHDL statements for verifying the project using a VHDL test bench, so that after simulation, a set of waveforms are obtained similar to the ones sketched in c).

h) Organise and plan the component *DataReg_4bit* as a FSM using the plan Y (*current_state* as a STD_LOGIC_VECTOR).

i) Which is the time resolution of the system? This is the minimum delay duration that can be programmed if the technology used is the CPLD Max II EPM2210F324C3 that has the following characteristics:

MAX II Device Features	**EPM2210**
t_{PD_gate} (ns)	1.7
t_{CO} (ns)	4.6



Present_state	**Sel_B**	**LDB**	**Sel_OP**	**LDW**	**Pulse**
Idle	0	0	10	0	0
Load_B	0	1	10	0	0
Load_B_&_W	1	1	01	1	0
Dec	0	0	01	1	0
Set_Output	0	0	10	0	1

Fig. 3
State diagram for the FSM in control of the system and the CC2 truth table to generate control and output signals.

Sel_OP	**Operation**
00	R ← A + B
01	R ← A - B
10	R ← B
11	R ← Shift left B

Fig. 4
The operations performed by the ALU included in the datapath circuit.

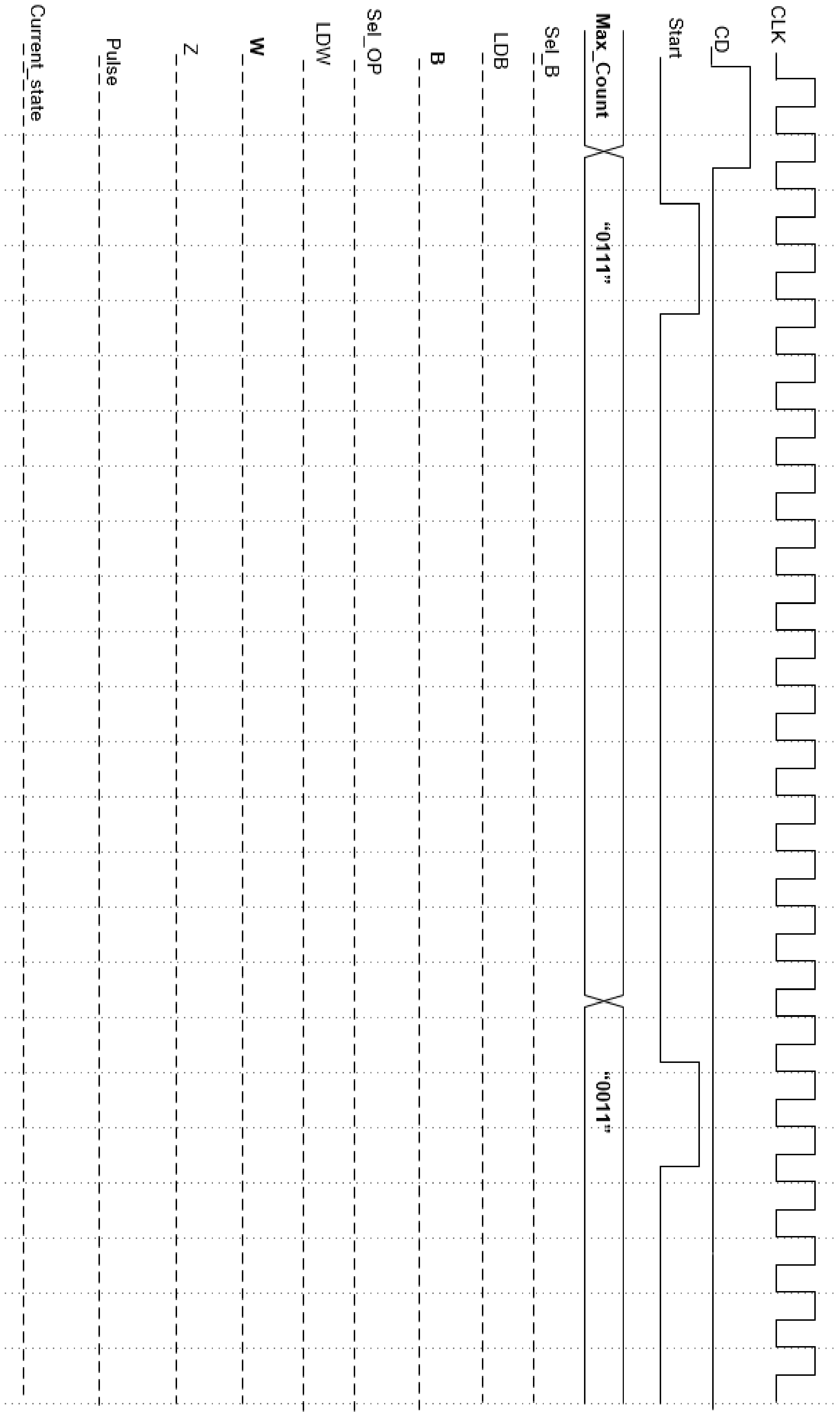


Fig. 5 Example sheet to draw the timing diagram of the outputs and the *current_state* variable for a given set of inputs.

Problem 2 (Ch3). Design.

We aim for comparison purposes, to redesign our delay generator from Problem 1 using a PIC18F4520 microcontroller and its Timer0 peripheral device. Fig. 6 shows the block diagram including the **Start** trigger, the **Delay** 8-bit vector which can be any value from 0 to 255, and the output **Pulse** which is the programmable delay signal and **LED_ON** to indicate that the circuit is on duty. See as well the example timing diagram. The duration of the delayed pulse generated is always of 20 µs.

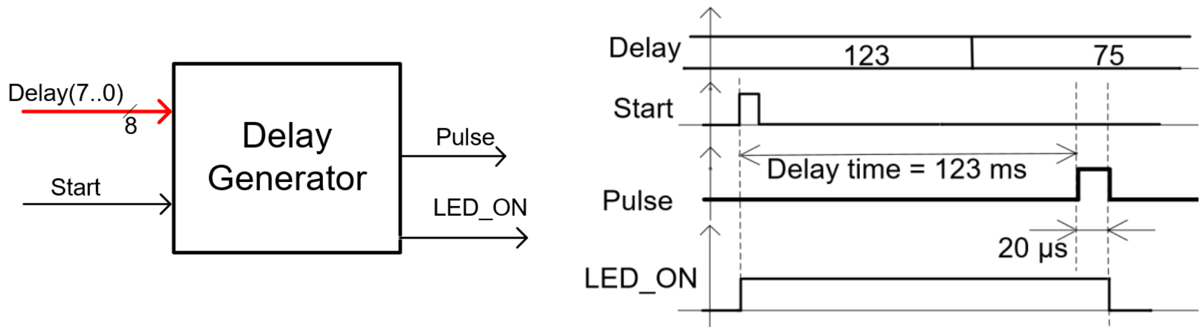


Fig. 6 Digital delay generator symbol and representative waveforms

a) Draw the schematic connecting the inputs and outputs to the PIC18F4520 as follows. Add the crystal oscillator of **16 MHz** and the MCLR_L circuits.
   Delay(7..6) → RC(4..3) ; **Delay**(5..3) → RD(4..2) ; **Delay**(2..0) → RB(7..5)
   Start → RB0 ; **Pulse** → RA4 ; **LED_ON** → RA3.

b) Draw the general configuration of the C program so that the system is organised as a FSM. Draw the hardware-software diagram representing inputs and output pins, RAM variables, and the most important functions: *read_inputs()*, *write_outputs()*, *output_logic()*, *state_logic()*, *ISR()*.

c) Explain how to configure the inputs and outputs in the *init_system()*. Explain how to allow interrupts from the external RB0/INT0 pin and the Timer0.

d) Explain how to implement the *read_inputs()* to generate a convenient *char* variable **Var_delay**.

e) Explain how to implement the *write_outputs()* function to send the **Var_Pulse** and the **Var_LED_ON** variables to the corresponding output pins.

f) Picture Fig. 7 shows the main hardware blocks of the Timer0 peripheral. Explain how to use it along with a software post-scaler RAM variable (**N3**) to generate the delay times for this application. Calculate the values N1, N2 and N3 to be able to program delays from 1 ms to 255 ms. Calculate the values N1 and N2 to program the 20 µs delay to generate the pulse.

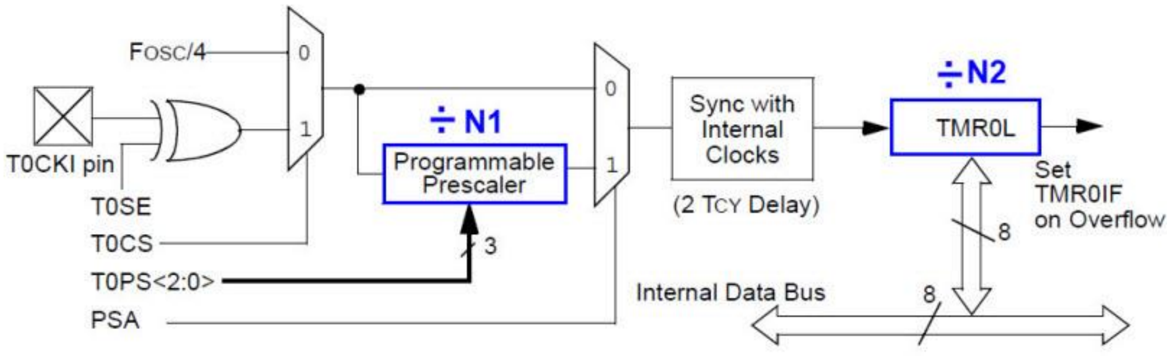


Fig. 7 Timer0 RTL equivalent circuit

g) Draw the state diagram for this application modifying the one presented in Fig. 3 so that the system is able to generate the pulse output with a programmable delay.

h) Draw the flow chart for the *ISR()* function.

i) Draw the truth table for the *output_logic()* and its flow chart ready for translating to C language.

j) Draw the truth table for the *state_logic()* and its flow chart ready for translating to C language.