ELEC3342: Digital System Designs
First Semester, 2018–19

Homework 1 (r1.2)

Due:

- Part (A) -- 19 Oct, 2018, 11:59pm
- Part (B) -- 19 Oct, 2018, 11:59pm

Instruction: Submit your answers electronically through Moodle.

There are 2 major parts in this homework. Part A includes questions that aim to help you with understanding the lecture materials. They resemble the kind of questions you will encounter in quizzes and the final exam. Some of these questions may also ask you to implement and test small designs in VHDL. This part of homework must be completed individually.

Part B of this homework contains a mini-project that you should work in groups of 2. Your submitted work will be graded by an auto tester and therefore you should make sure your submitted files conform to the required format.

The following summarize the 2 parts.

Part	Type	Indv/Grp
A	Basic problem set	Individual
B	Mini-project	Group of 2

In all cases, you are encouraged to discuss the homework problems offline or online using Piazza. However, you should not ask for or give out solution directly as that defeat the idea of having homework exercise. Giving out answers or copying answers directly will likely constitute an act of plagiarism, which is a serious offense.

Part A: Problem Set

A.1

A.1.1 For each of the truth table in Figure A.1, do the following:

(i) Draw the corresponding K-map

(ii) Minimize using K-map and write the resulting Boolean equation in sum-of-products canonical form;

(iii) Design a simple circuit corresponding to the truth table. You may use any gate with any number of inputs.

(iv) Design a simple circuit using only 2-input NAND, NOR, NOT gates.

A.1.2 For each of the truth table in Figure A.1, implement your resulting circuit with VHDL. Your submitted VHDL must meet the following requirements:

- The entity of your circuit should be named as:

  `A_1_<truth_table_number>`

- Each circuit should have a top-level VHDL entity with the corresponding port signal.
- All ports must be of type `STD_LOGIC`
- Save your vhdl file as:

  `A_1_<truth_table_number>.vhdl`

Submit your VHDL files.

A.2

A.2.1 Simplify the following Boolean equations using Boolean theorems. Check for correctness using a truth table or K-map.

(a) $Y = AC + \overline{A}\,\overline{B}\,C$

(b) $Y = \overline{A}, \overline{B} + \overline{A}\,B\,\overline{C} + \overline{(A + \overline{C})}$

(c) $Y = \overline{A}\,\overline{B}\,\overline{C}\,\overline{D} + A\overline{B}\,\overline{C} + A\overline{B}C\overline{D} + ABD + \overline{A}\,\overline{B}C\overline{D} + B\overline{C}D + \overline{A}$

(1.)

A	B	Y
0	0	0
0	1	0
1	0	1
1	1	0

(2.)

A	B	C	Y
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	0

(3.)

A	B	C	D	Y
0	0	0	0	0
0	0	0	1	1
0	0	1	0	0
0	0	1	1	1
0	1	0	0	1
0	1	0	1	0
0	1	1	0	0
0	1	1	1	1
1	0	0	0	0
1	0	0	1	1
1	0	1	0	0
1	0	1	1	0
1	1	0	0	0
1	1	0	1	1
1	1	1	0	0
1	1	1	1	1

(4.)

A	B	C	D	E	Y
0	0	0	0	0	0
0	0	0	0	1	0
0	0	0	1	0	0
0	0	0	1	1	0
0	0	1	0	0	1
0	0	1	0	1	0
0	0	1	1	0	1
0	0	1	1	1	1
0	1	0	0	0	0
0	1	0	0	1	1
0	1	0	1	0	1
0	1	0	1	1	1
0	1	1	0	0	0
0	1	1	0	1	1
0	1	1	1	0	0
0	1	1	1	1	1
1	0	0	0	0	0
1	0	0	0	1	1
1	0	0	1	0	0
1	0	0	1	1	1
1	0	1	0	0	1
1	0	1	0	1	0
1	0	1	1	0	1
1	0	1	1	1	1
1	1	0	0	0	0
1	1	0	0	1	1
1	1	0	1	0	1
1	1	0	1	1	1
1	1	1	0	0	0
1	1	1	0	1	0
1	1	1	1	0	0
1	1	1	1	1	0

Figure A.1: Truth Tables for questions Question A.1.1 and Question A.1.2

A.3

Using De Morgan equivalent gates and bubble pushing methods, redaw the circuit in Figure A.2 so that there is only AND, OR gates with optional bubble input. [Based on DDCA Exercise 2.27]

A.4

A circuit has four inputs and two outputs. The inputs `A[3:0]` represent a number from 0 to 15. Output `P` should be TRUE if the number is prime (0 and 1 are not prime, but 2, 3, 5, and so on, are prime). Output `D` should be TRUE if the number is divisible by 3. Give simplified Boolean equations for each output and sketch a circuit. [DDCA 2.35]

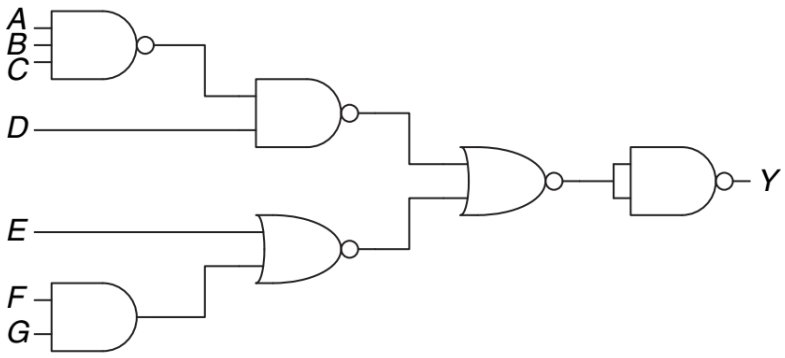

Figure A.2: Circuit for schematic (Figure 2.84 from DDCA)

A.5

An M-bit *thermometer code* for the number k consists of k `1`'s in the least significant bit positions and $M–k$ `0`'s in all the more significant bit positions. A binary-to-thermometer code converter has N inputs and $2^N - 1$ outputs. It produces a $2^N - 1$ bit thermometer code for the number specified by the input. For example, if the input is `110`, the output should be `0111111`. Design a $3:7$ binary-to-thermometer code converter. Give a simplified Boolean equation for each output, and sketch a schematic. [DDCA 2.38]

A.6

Determien the propagation delay and contamination delay of the circuit in Figure A.2. Use the gate delay from Figure A.3

Gate	t_{pd} (ps)	t_{cd} (ps)
NOT	15	10
2-input NAND	20	15
3-input NAND	30	25
2-input NOR	30	25
3-input NOR	45	35
2-input AND	30	25
3-input AND	40	30
2-input OR	40	30
3-input OR	55	45
2-input XOR	60	40

Figure A.3: Gate Delay [DDCA Table 2.8]

A.7

Implement the circuit of Figure A.2 in VHDL. Implement the following the circuit with the following different style and submit the resulting VHDL file with the given name.

(a) `A_7_a.vhdl`: As a single assignment statement in the architecture definition.

(b) `A_7_b.vhdl`: As multiple assignment statements in the architecture definition without using a `process` statement. Define additional signals as needed.

(c) `A_7_c.vhdl`: As a single assignment statement inside a `process` statement in the architecture definition. Include all input in the process sensitivity list.

(d) `A_7_d.vhdl`: As multiple assignment statements inside a `process` statement in the architecture definition. Define additional variables as needed. Include all input signals in the process sensitivity list.

(e) `A_7_e.vhdl`: Similar to (c) above, except that only signal `A`, `B` are in the process sensitivity list.

For each implementation, also submit a screen capture of the generated schematic from Xilinx Vivado tools. Are the produced circuit schematics the same in all above cases? If not, what are the differences?

A.8

Given the input waveform shown in Figure A.4, sketch the output, Q, of an SR latch.

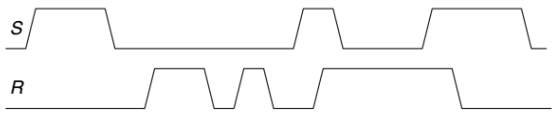

Figure A.4: Input waveform for SR latch [DDCA Fig 3.61]

A.9

Given the input waveform shown in Figure A.5, sketch the output, Q, of:

(a) D latch

(b) D flip flop

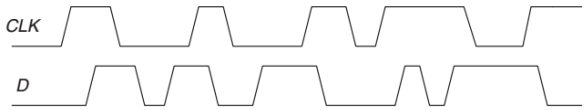

Figure A.5: Input waveform for D latch and D flip flop [DDCA Fig 3.64]

A.10

An edge-triggered *JK flip-flop* functions similarly to an SR latch. A JK flip-flop has 2 inputs, `J`, `K`, and a clock signal `clk`. On the rising edge of the clock, it updates its output `Q` according to the following rules:

- If `J` and `K` are both `0`, `Q` remains its old value;
- If `J` is `1` and `K` is `0`, then `Q` becomes `1`;
- If `K` is `1` and `J` is `0`, then `Q` becomes `0`;
- If both `K` and `J` are `1`, then `Q` toggles (`1` becomes `0`, `0` becomes `1`).

Perform the following tasks:

(a) Design a JK flip flop using a D flip-flop and some combinational logic.

(b) Implement your JK flip flop in VHDL. Submit your resulting file as `A_jk.vhdl`

A.11

Describe in words what the state machine in Figure A.6 does. Is this a Mealy machine or a Moore machine?

Using binary state encodings, complete the next state logic and output logic table for the FSM. Write Boolean equations for the next stage and output logic.

Implement this FSM in vhdl using 3 separated processes, one for state registeres, one for next state logic and one for output logic. Submit your FSM as `A_11_fsm.vhdl`.

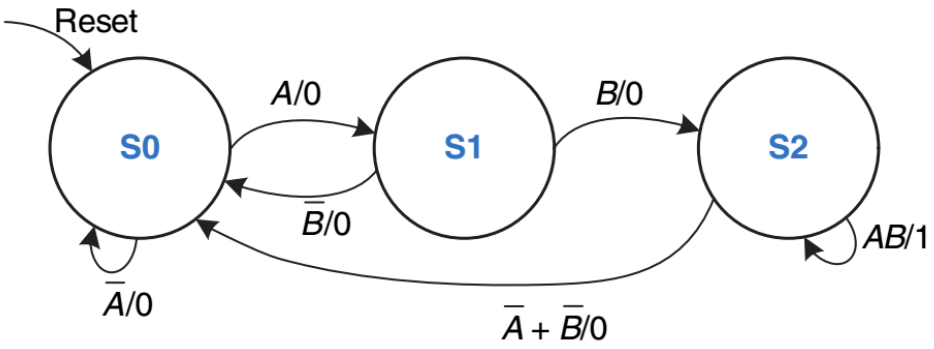

Figure A.6: State transition diagrm [DDCA Fig. 3.70]

Part B: Mini-Project

B.1 Morse Code Decoder

This semester you will be building a complete Morse Code decoding system. In this homework, you will begin your design with the core decoder finite state machine (FSM).

B.1.1 Morse Code Introduction Originally developed in the 19th century, Morse code has become one of the most widely used communication coding system even to this day. Morse code is a relatively simple system that uses a series of on-off signals to transmit text information (letters and numerals). These on-off signals can be transmit as telegram, sound, or even light.

The key differentiating factor is the duration of the signal. A long signal is called a 'dash', while a short signal is called a 'dot' in Morse code convention. Figure B.1 shows a chart of the standard international Morse code that you will adopt in this homework.

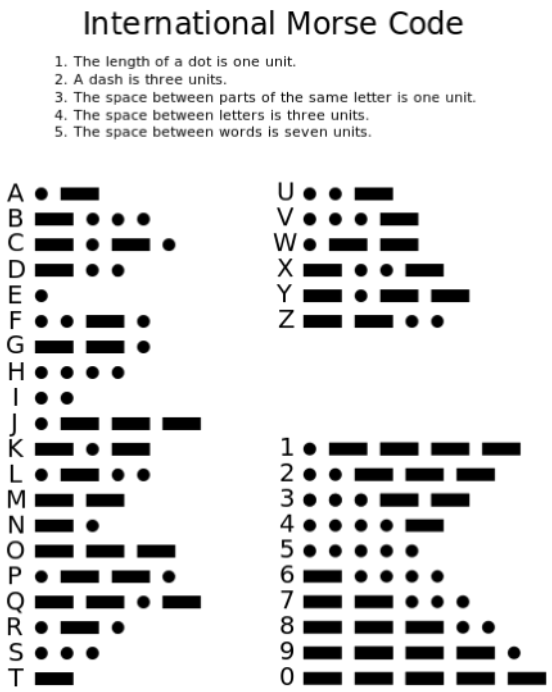

Figure B.1: Chart of International Morse Code

Furthermore, the duration of the time between a 'dot' and a 'dash' (a signal gap), the time between 2 letters (letter gap), and the duration between 2 words (word gap) are also important and are well

defined. Using the length of a 'dot' as a unit, the following table summaries the relative duration of each part of the code.

dot	1
dash	3
signal gap	1
letter gap	3
word gap	7

Although the relative duration of various parts of the code are well defined, the actual length of a 'dot' may vary. Furthermore, in practical applications, due to human error and other factors, the relative length of the signals and gaps may vary slightly as well.

B.1.2 Task 1: Morse Code Decoder Your main task for this homework is to develop the core morse code decoder of the system. Figure B.2 shows a block diagram of your decoder.

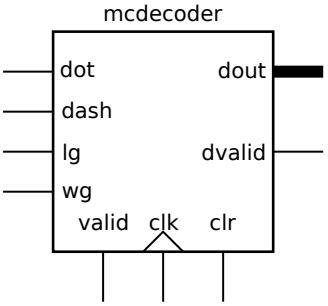

Figure B.2: Block diagram of your Morse code decoder

Your decoder has the following input and output port:

Port Name	Dir	Width	Description
`dot`	in	1	'1' when input is a dot, '0' otherwise
`dash`	in	1	'1' when input is a dash, '0' otherwise
`lg`	in	1	'1' when input is a letter gap, '0' otherwise
`wg`	in	1	'1' when input is a word gap, '0' otherwise
`valid`	in	1	When `valid` is '0' your decoder should ignore the input `dot`, `dash`, `lg` and `wg`. Your decoder should only respond to the above input when `valid` is '1'.
`clr`	in	1	When asserted ('1'), the entire decoder should reset to initial state regardless of the value of `clk`.
`clk`	in	1	The main clock to your module
`dout`	out	8	Detected symbol encoded as 8-bit ASCII code
`dvalid`	out	1	Asserted when `dout` is valid. '0' otherwise

B.1.3 Example Run The waveform shown in Figure B.3 illustrates the expected behavior of your block when the input is the sequence "AT M" (A, T, space, M).

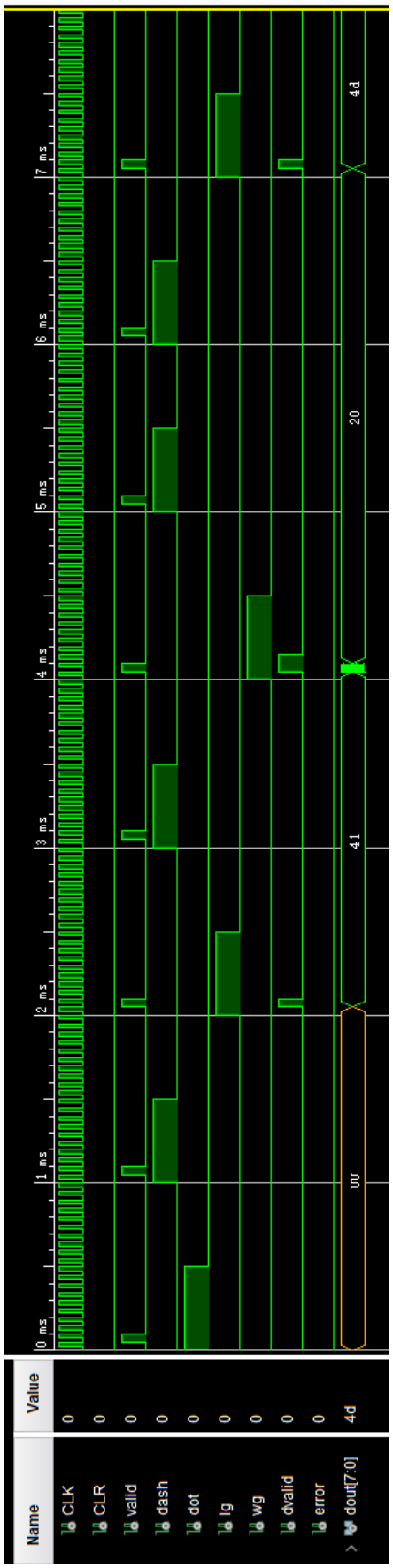

Figure B.3: Sample waveform when the input is the sequence A, T, space, M. Note that mcdecoder ignores its input when `valid` is low.

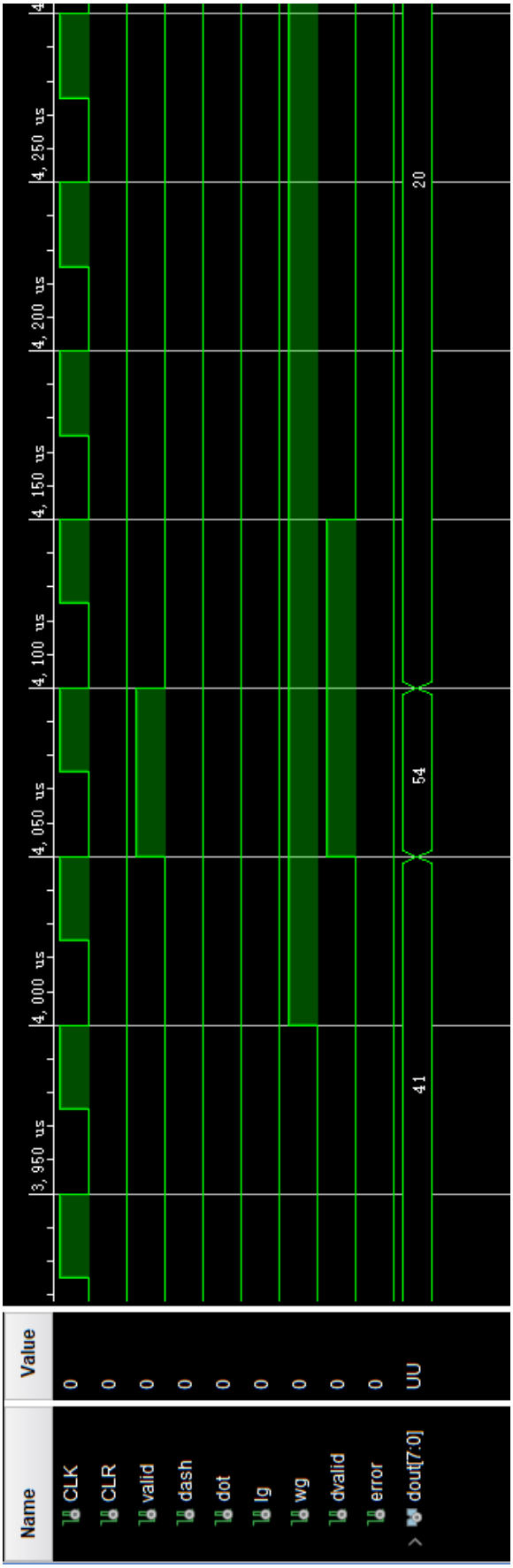

Figure B.4: Zoom into the area near T and space.

From the waveform, you will notice that your decoder reads in an input whenever `valid` is '1'. Once a `lg` or a `wg` is encountered, the correct symbol is determined from the previously received `dot`s and `dash`es.

Furthermore, note that if a `wg` is received, your decoder should also output an additional space symbol (ASCII code 0x20).

B.1.4 Submission For this part of homework, you should submit one file `mcdecoder.vhd` through Moodle. A template for that file is available on the course website, together with a sample testbench.