TABLE OF CONTENTS

CALCULATIONS

Section 1.1: Display Mode 0 calculations

According to the specifications, there appear to be 7 states (one for the 6 segments in the perimeter of the 7-segment display, and 1 for no segments displayed), which can be represented with the use of $\lceil \log_2 7 \rceil = 3$ flip-flops. Let these 3 flip-flops be D flip-flops, as they are easier to work with, labelled $D0, D1$, and $D2$, with respective outputs $Q0$, $Q1$, $Q2$. Note that for the remainder of the documentation, all flip-flops used are D flip-flops.

Table 1. Mapping of states of Display Mode 0

State	$Q0$	$Q1$	$Q2$
$S0$	0	0	0
$S1$	0	0	1
$S2$	0	1	0
$S3$	0	1	1
$S4$	1	0	0
$S5$	1	0	1
$S6$	1	1	0

At any given time, there are also 6 relevant segments as output: a, b, c, d, e, and f. However, because the state of Display Mode 1 is affected by Display Mode 0, there must be a signal between the two display modes. This segment can be the segment a, as the transition of a from 1 to 0 can represent the transition from $S6$ to $S0$. Using a as a signal is discussed in greater depth in Section 1.2.

Using a as the signal between the two display modes allows the circuit to be represented as a Moore circuit, because the output is no longer bound to the transition; rather, the transition is caught by the flipping of a. There also exists a separate input, a button X, which dictates when the circuit can change states. The Moore circuit is illustrated in Figure 1.

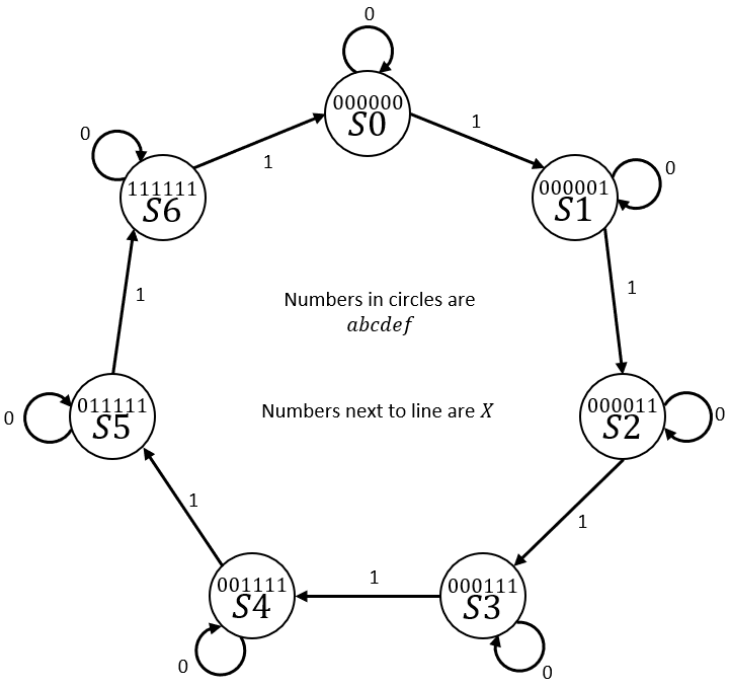

Figure 1. Moore state diagram

Table 2. Moore state table

Current state	$X = 0$	$X = 1$	a	b	c	d	e	f
$S0$	$S0$	$S1$	0	0	0	0	0	0
$S1$	$S1$	$S2$	0	0	0	0	0	1
$S2$	$S2$	$S3$	0	0	0	0	1	1
$S3$	$S3$	$S4$	0	0	0	1	1	1
$S4$	$S4$	$S5$	0	0	1	1	1	1
$S5$	$S5$	$S6$	0	1	1	1	1	1
$S6$	$S6$	$S0$	1	1	1	1	1	1

This Moore state diagram has the following truth table:

Table 3. Truth table of the circuit

Primary input	**Current state**			**Next state/Secondary input**			**Primary outputs**					
X	$Q0$	$Q1$	$Q2$	$Q0^+/D0$	$Q1^+/D1$	$Q2^+/D2$	a	b	c	d	e	f
0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	1	0	0	1	0	0	0	0	0	1
0	0	1	0	0	1	0	0	0	0	0	1	1
0	0	1	1	0	1	1	0	0	0	1	1	1
0	1	0	0	1	0	0	0	0	1	1	1	1
0	1	0	1	1	0	1	0	1	1	1	1	1
0	1	1	0	1	1	0	1	1	1	1	1	1
0	1	1	1	X	X	X	X	X	X	X	X	X
1	0	0	0	0	0	1	0	0	0	0	0	0
1	0	0	1	0	1	0	0	0	0	0	0	1
1	0	1	0	0	1	1	0	0	0	0	1	1
1	0	1	1	1	0	0	0	0	0	1	1	1
1	1	0	0	1	0	1	0	0	1	1	1	1
1	1	0	1	1	1	0	0	1	1	1	1	1
1	1	1	0	0	0	0	1	1	1	1	1	1
1	1	1	1	X	X	X	X	X	X	X	X	X

Table 4. Two-level forms for Display Mode 1 components

Component	**Two-level form (SOP)**
$D0$	$Q0Q1' + X'Q0 + XQ1Q2$
$D1$	$XQ1'Q2 + X'Q1 + Q0'Q1Q2'$
$D2$	$X'Q2 + XQ1'Q2' + XQ0'Q2'$
a	$Q0Q1$
b	$Q0Q2 + Q0Q1$
c	$Q0$
d	$Q0 + Q1Q2$
e	$Q0 + Q1$
f	$Q0 + Q2 + Q1$

Associated K-maps can be found under Section 4.1.

More information regarding the use of a Mealy circuit instead can be found under Remarks.

Section 1.2: Display Mode 1's counter

According to the specifications, there are 2 primary purposes of Display Mode 1: it must contain a method of keeping track of the number of cycles done by Display Mode 0 and function as a method of stopping any further interaction with the system should the number of cycles completed reach 10.

Keeping track of the number of cycles comes with its own problems. As established in Section 1.1, a will be used as the signal between the two modes. However, there is a 1 clock-cycle delay between the transition from $S6$ to $S0$ and Display Mode 1 updating as established in the forum. This means that after a transitions from 1 to 0 another clock cycle must first pass before Display Mode 1 properly updates. There are three problems to solve: firstly, catching the transition; secondly, waiting a clock cycle; and lastly, appending a counter.

The easiest of the three problems is the last. A counter which counts up is fairly easy to implement, being tackled in a previous work, Lab Exercise 9. Because Display Mode 1 must be capable of counting up to 10, the counter must accommodate 4-bits, which requires 4 flip-flops. For simplicity, the first flip-flop of this four flip-flop system will be referred to as $counter$ for the remainder of this documentation, while the counting system as whole will be referred to as counter.

The first problem of catching the transition can be solved by connecting $\bar{a}$ to the CLK input of $counter$. This makes it so the counter will count up only if a goes from 1 to 0 (staying in 1 or 0 or going from 0 to 1 will not append the counter).

The second problem, delaying the counter by 1 clock cycle, is achieved by adding one more flip-flop, which will be referred to as $delay$. Note that when n flip-flops are used in series, a signal can be delayed by n clock cycles. By removing $\bar{a}$ from $counter$ and instead connecting a to the D of $delay$, and connecting the $\bar{Q}$ of $delay$ to the D of $counter$, the signal from a is delayed by one clock cycle. Note the switch from $\bar{a}$ to a, as $\bar{Q}$ serves the same function as a NOT gate attached to a.

Section 1.3: Stopping the system

Display Mode 1 must also be capable of stopping the entire system should it reach the count of 10. This can be achieved by another output, referred to as $STOP$, which is HIGH when Display Mode 1 counts to 10. This is very simple, as 10 is dictated by two bits in a 4-bit number, thus $STOP$ is just those two bits connected by an AND gate (note that the counter should stop appending beyond 10, so any other number with those two bits also HIGH is irrelevant).

$STOP$ serves two purposes: firstly, to prevent the counter from appending any further, and secondly, to lock the system to Display Mode 1. The former can be achieved through a series of gates connecting $STOP$ and a, whose output will become the D input of $delay$. The latter is discussed in Section 1.4.

Table 5. Truth table of the D input for $delay$

$STOP$	a	D input
0	0	0
0	1	1
1	0	0
1	1	0

The truth table in Table 5 can be implemented through a single gate: $\overline{STOP}$ AND a. Because the D-input of $delay$ remains indefinitely LOW should $STOP$ be HIGH, the counter will stop appending.

In total, Display Mode 1 has 5 flip-flops (four for the counter and 1 for $delay$), as well as 4 outputs (4 bits for the counter, and 1 for $STOP$), and 2 gates.

Section 1.4: System integration

Note that the outputs for Display Mode 1 are in binary bits ($ABCD$, where A is the MSB), which must first be decoded into bits for the 7-segment display. Unfortunately, none of the 7-segment-display-driver provided by Logisim, Logisim-ITA's built-in 7-segment display decoder, nor 7448 IC have the correct display for the numbers 9 and 6 as indicated in the specifications, so a new circuit must be created. As previous implementations can be lifted, modifications to the output of Lab Exercise 5 can be utilized to decrease the added workload.

Table 6. Two-level forms for the binary to 7-segment decoder

Component	Two-level form (SOP)
a	$(B' + C + D)(A + B + C + D')$
b	$(B' + C + D')(B' + C' + D)$
c	$A + B + C' + D$
d	$(B' + C + D)(A + B + C + D')(B' + C' + D')(A' + C')$
e	$(B' + C)(D')$
f	$(A + B + D')(C' + D')(A + B + C')$
g	$(A + B + C)(B' + C' + D')$

Associated calculations can be found under Section 4.2.

There also remains the requirement of switching between the two display modes. This can be achieved through a series of 2-bit multiplexers whose address bit is the switch. There exists one multiplexer for every segment except g, because Display Mode 0 does not have a g output; instead, an AND gate between the switch and g can be used to assure that g will not turn on if it is not needed.

To use the switch as the address of a 2-bit multiplexer, the switch must toggle between HIGH and LOW, which Logisim-ITA's basic switch does not do. This can be remedied by connecting the switch to either a Power or constant input and connecting its output to a pull resistor. Power, constant inputs, and pull resistors are deemed valid for use according to the forum.

There also remains the functionality of Display Mode 1's $STOP$ output to lock the system to Display Mode 1. This is achieved through an OR gate between the switch and $STOP$, which guarantees that Display Mode 1 is always active if $STOP$ is HIGH.

IMPLEMENTATION

Section 2.1: Display Mode 0 implementation

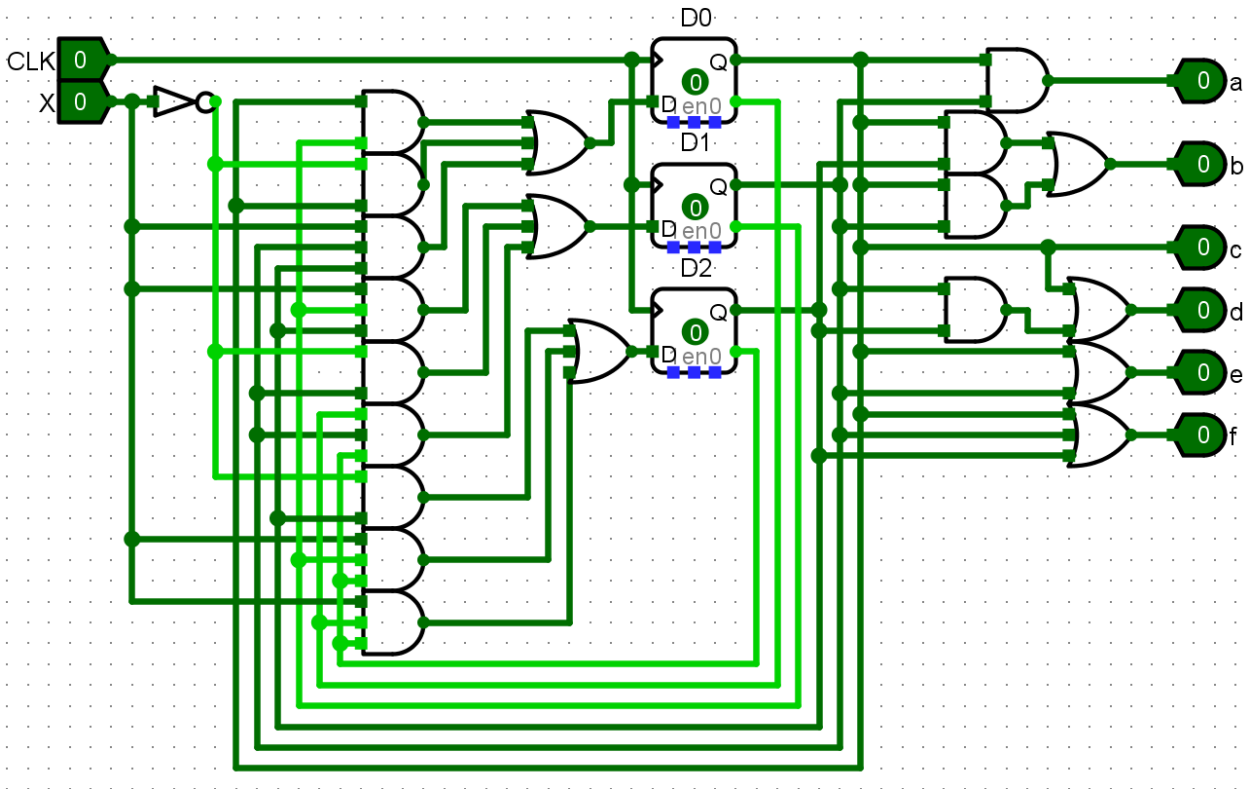

Figure 2. The Display Mode 0 subcircuit

As pictured in Figure 2, Display Mode 0 is implemented through a subcircuit which realizes the circuit diagram in Figure 1 and the two-level forms in Table 4. There is a global clock for both Display Mode 0 and Display Mode 1, thus CLK must be an input. The input X is the system's button, while $a, b, c, d, e,$ and f refer to segments in the 7-segment display.

Section 2.2. Display Mode 1 implementation

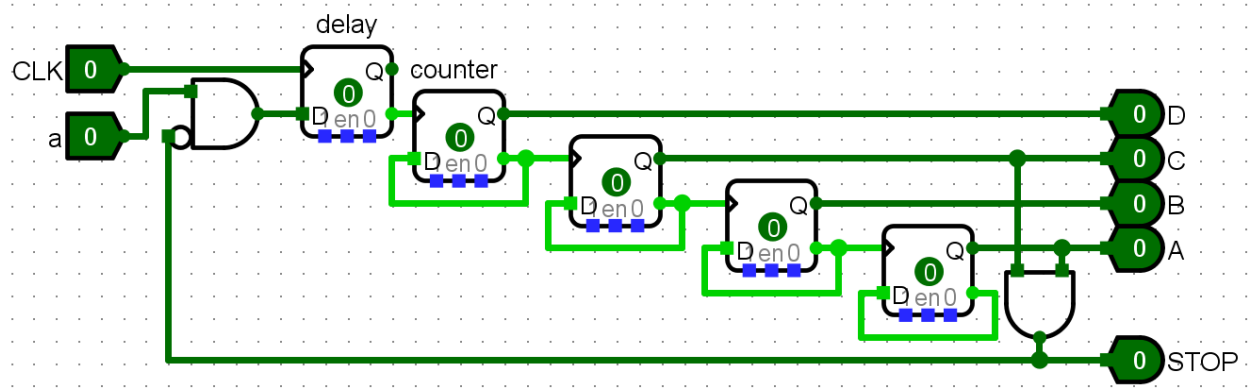

Figure 3. The Display Mode 1 subcircuit

Figure 3 illustrates the subcircuit for Display Mode 1, which implements the entirety of Sections 1.2 and 1.3. Pictured are the five flip-flops, the $STOP$ output, the four output bits $A, B, C,$ and D (where A is the MSB), as well as the two AND gates.

Section 2.3. Utility subcircuits implementation

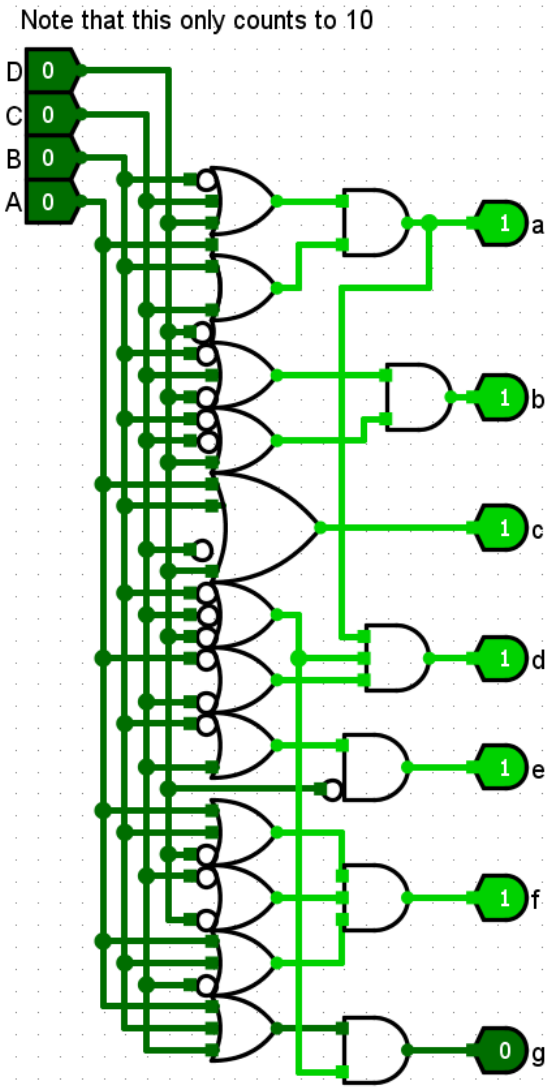

Figure 4. The binary to 7-segment decoder subcircuit

The subcircuit depicted in Figure 4 implements the two-level forms seen in Table 6.

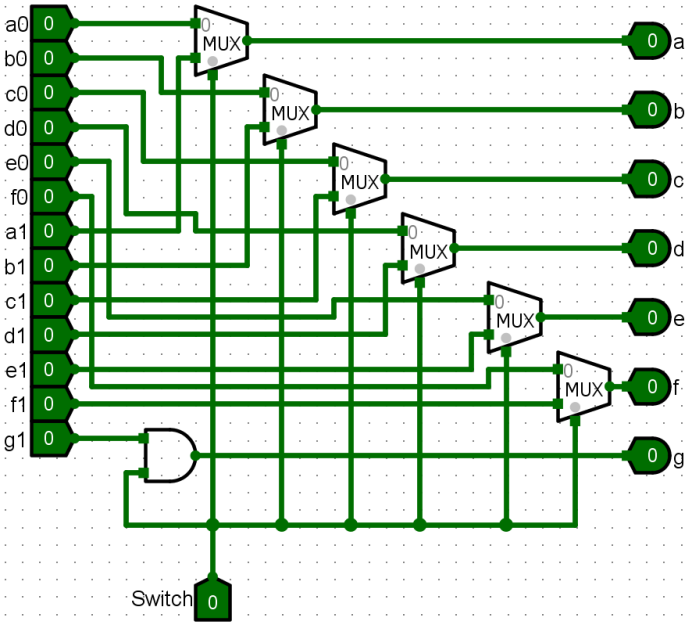

Figure 5. The switcher subcircuit

Figure 5 illustrates the 2-bit multiplexers described in Section 1.4. The inputs $a0, b0, c0, d0, e0$, and $f0$ refer to the outputs of Display Mode 0 which are set to address 0, while $a1, b1, c1, d1, e1, f1$, and $g1$ refer to the outputs of Display Mode 1 which are set to address 1. When $Switch$ is HIGH, only address 1 is used as outputs, but when $Switch$ is LOW, address 0 is used instead. Note that $g1$ has a gate instead of a multiplexer, as described prior.

Section 2.4. High-level implementation

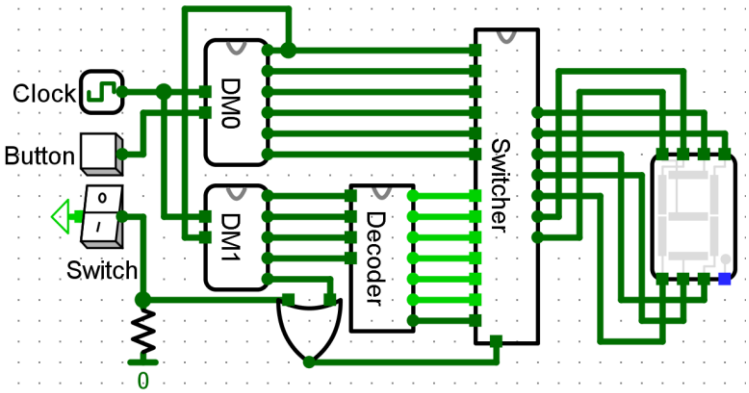

Figure 6. Implementation of the entire system

Figure 6 depicts the whole functional system. It has three inputs, the clock, a button, and a switch, and one output, a 7-segment display. The subcircuits for Display Mode 0 and Display Mode 1 (labelled as DM0 and DM1 respectively) are both connected to a common clock, while the button is connected to the X input of DM0. Also pictured are the pull resistor and power components mentioned in Section 1.4. Note that the a output of DM0 is connected to the a input of DM1, and that there exists an OR gate between the switch and the $STOP$ output of DM1. DM1 is connected to the decoder in Figure 4, while the decoder and DM0 are both connected to the switcher seen in Figure 5. The switcher outputs segments a to g to the 7-segment display.

REMARKS

While a Moore implementation was used in this documentation, there is a way to use a Mealy circuit instead. Note that Display Mode 1 should only increase if Display Mode 0 transitions from S6 to S0, and not necessarily when Display Mode 0 is at $S0$. A Mealy circuit can be used if another output is added whose sole purpose is to serve as a signal between the two display modes. This output cannot be represented in a Moore circuit because it is the transition which dictates its value, not the state.

There may exist a method of implementing the switch without the use of a pull resistor; that is, a method which takes advantage of the switch's floating output. However, using multiplexers was the most straightforward solution at the time of writing, and a pull resistor and power component are an easy workaround.

The components, as seen under Implementation, in the project2.circ file, and in the video presentation, may not look the same as they do by default in Logisim-ITA. This is because the project was originally implemented in Logisim 2.7.1, then imported to Logisim-ITA.

Further optimizations to the system (less gates, less components, etc.). may exist but are not discussed in this documentation.

Video presentation can be found at
https://drive.google.com/file/d/18Nt1xhvMSKJGPaB3wjAzp79y92QVfOXn/view?usp=sharing

APPENDIX

Section 4.1: K-maps for Display Mode 0 calculations

Table 7. K-map for $D0$

		$Q1Q2$			
		00	01	11	10
$XQ0$	00	0	0	0	0
	01	1	1	X	1
	11	1	1	X	0
	10	0	0	1	0

$SOP: Q0Q1' + X'Q0 + XQ1Q2$

Grey, orange, blue box respectively

Table 8. K-map for $D1$

		$Q1Q2$			
		00	01	11	10
$XQ0$	00	0	0	1	1
	01	0	0	X	1
	11	0	1	X	0
	10	0	1	0	1

$SOP: XQ1'Q2 + X'Q1 + Q0'Q1Q2'$

Grey, orange, blue box respectively

Table 9. K-map for $D2$

		$Q1Q2$			
		00	01	11	10
$XQ0$	00	0	1	1	0
	01	0	1	X	0
	11	1	0	X	0
	10	1	0	0	1

$SOP: X'Q2 + XQ1'Q2' + XQ0'Q2'$

Grey, orange, blue box respectively

Note that the values for $a, b, c, d, e,$ and f are not dependent on X, rather they only depend on $Q0, Q1,$ and $Q2$, as is the nature of Moore circuits.

Table 10. K-map for a

		$Q1Q2$			
		00	01	11	10
$Q0$	0	0	0	0	0
	1	0	0	X	1

$SOP: Q0Q1$

Table 11. K-map for b

		$Q1Q2$			
		00	01	11	10
$Q0$	0	0	0	0	0
	1	0	1	X	1

$SOP: Q0Q2 + Q0Q1$

Grey, orange box respectively

Table 12. K-map for *c*

		$Q1Q2$			
		00	01	11	10
$Q0$	0	0	0	0	0
	1	1	1	X	1

$SOP: Q0$

Table 13. K-map for *d*

		$Q1Q2$			
		00	01	11	10
$Q0$	0	0	0	1	0
	1	1	1	X	1

$SOP: Q0 + Q1Q2$

Grey, orange box respectively

Table 14. K-map for *e*

		$Q1Q2$			
		00	01	11	10
$Q0$	0	0	0	1	1
	1	1	1	X	1

$SOP: Q0 + Q1$

Grey, orange box respectively

Table 15. K-map for *f*

			$Q1Q2$			
			00	01	11	10
$Q0$	0		0	1	1	1
	1		1	1	X	1

$SOP: Q0 + Q2 + Q1$

Grey, orange, blue box respectively

Section 4.2: Calculations for the binary to 7-segment display decoder

Table 16. Truth table of the binary to 7-segment display decoder

A	B	C	D	a	b	c	d	e	f	g
0	0	0	0	1	1	1	1	1	1	0
0	0	0	1	0	1	1	0	0	0	0
0	0	1	0	1	1	0	1	1	0	1
0	0	1	1	1	1	1	1	0	0	1
0	1	0	0	0	1	1	0	0	1	1
0	1	0	1	1	0	1	1	0	1	1
0	1	1	0	1	0	1	1	1	1	1
0	1	1	1	1	1	1	0	0	0	0
1	0	0	0	1	1	1	1	1	1	1
1	0	0	1	1	1	1	1	0	1	1
1	0	1	0	1	1	1	0	1	1	1
1	0	1	1	X	X	X	X	X	X	X
1	1	0	0	X	X	X	X	X	X	X
1	1	0	1	X	X	X	X	X	X	X
1	1	1	0	X	X	X	X	X	X	X
1	1	1	1	X	X	X	X	X	X	X

The binary number is formatted as $ABCD$, where A is the MSB.

Table 17. K-map for a

		CD			
		00	01	11	10
AB	00	1	0	1	1
	01	0	1	1	1
	11	X	X	X	X
	10	1	1	X	1

$POS: (B' + C + D)(A + B + C + D')$

Groupings from left to right

Table 18. K-map for b

		CD			
		00	01	11	10
AB	00	1	1	1	1
	01	1	0	1	0
	11	X	X	X	X
	10	1	1	X	1

$POS: (B' + C + D')(B' + C' + D)$

Groupings from left to right

Table 19. K-map for c

		CD			
		00	01	11	10
AB	00	1	1	1	0
	01	1	1	1	1
	11	X	X	X	X
	10	1	1	X	1

$POS: A + B + C' + D$

Table 20. K-map for d

		CD			
		00	01	11	10
AB	00	1	0	1	1
	01	0	1	0	1
	11	X	X	X	X
	10	1	1	X	0

$POS: (B' + C + D)(A + B + C + D')(B' + C' + D')(A' + C')$

Groupings from left to right

Table 21. K-map for e

		CD			
		00	01	11	10
AB	00	1	0	0	1
	01	0	0	0	1
	11	X	X	X	X
	10	1	0	X	1

$POS: (B' + C)(D')$

Blue box, orange box respectively

Table 22. K-map for f

		CD			
		00	01	11	10
AB	00	1	0	0	0
	01	1	1	0	1
	11	X	X	X	X
	10	1	1	X	1

$POS: (A + B + D')(C' + D')(A + B + C')$

Green box, blue box, orange box respectively

Table 23. K-map for g

		CD			
		00	01	11	10
AB	00	0	0	1	1
	01	1	1	0	1
	11	X	X	X	X
	10	1	1	X	1

$POS: (A + B + C)(B' + C' + D')$

Blue box, orange box respectively