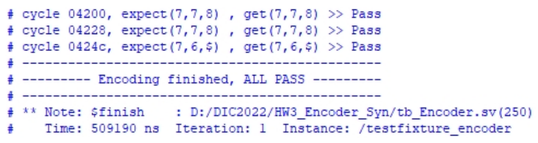
2022 Digital IC Design Homework 3

NAME	王梓帆
Student ID	N26114976

Simulation Result

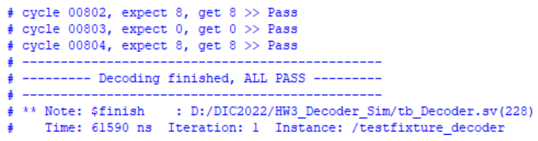
Functional simulation	Pass (encoder)	Pass (decoder)	Gate-level simulation	Pass (encoder)	Pass (decoder)

(your pre-sim result) encoder img0



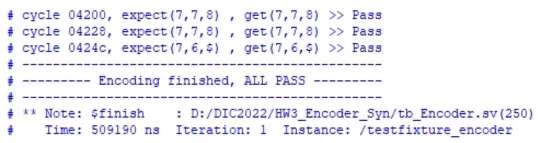
```
# cycle 04200, expect(7,7,8) , get(7,7,8) >> Pass
# cycle 04228, expect(7,7,8) , get(7,7,8) >> Pass
# cycle 0424c, expect(7,6,$) , get(7,6,$) >> Pass
# -----------------------------------------------
# --------- Encoding finished, ALL PASS ---------
# -----------------------------------------------
# ** Note: $finish    : D:/DIC2022/HW3_Encoder_Syn/tb_Encoder.sv(250)
#    Time: 509190 ns  Iteration: 1  Instance: /testfixture_encoder
```

(your pre-sim result) decoder img0



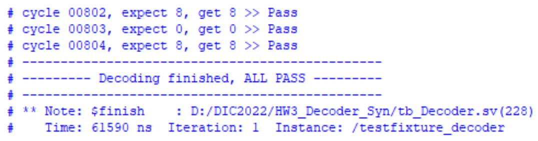
```
# cycle 00802, expect 8, get 8 >> Pass
# cycle 00803, expect 0, get 0 >> Pass
# cycle 00804, expect 8, get 8 >> Pass
# -----------------------------------------------
# --------- Decoding finished, ALL PASS ---------
# -----------------------------------------------
# ** Note: $finish    : D:/DIC2022/HW3_Decoder_Sim/tb_Decoder.sv(228)
#    Time: 61590 ns  Iteration: 1  Instance: /testfixture_decoder
```

(your post-sim result) encoder img0



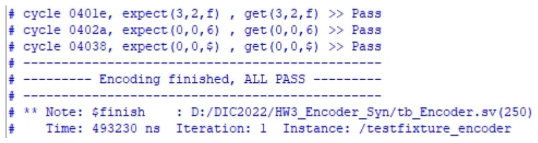
```
# cycle 04200, expect(7,7,8) , get(7,7,8) >> Pass
# cycle 04228, expect(7,7,8) , get(7,7,8) >> Pass
# cycle 0424c, expect(7,6,$) , get(7,6,$) >> Pass
# -----------------------------------------------
# --------- Encoding finished, ALL PASS ---------
# -----------------------------------------------
# ** Note: $finish    : D:/DIC2022/HW3_Encoder_Syn/tb_Encoder.sv(250)
#    Time: 509190 ns  Iteration: 1  Instance: /testfixture_encoder
```

(your post-sim result) decoder img0



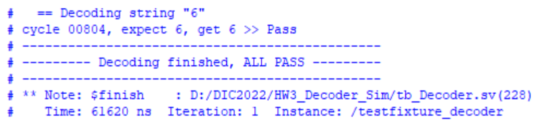
```
# cycle 00802, expect 8, get 8 >> Pass
# cycle 00803, expect 0, get 0 >> Pass
# cycle 00804, expect 8, get 8 >> Pass
# -----------------------------------------------
# --------- Decoding finished, ALL PASS ---------
# -----------------------------------------------
# ** Note: $finish    : D:/DIC2022/HW3_Decoder_Syn/tb_Decoder.sv(228)
#    Time: 61590 ns  Iteration: 1  Instance: /testfixture_decoder
```

(your pre-sim result) encoder img1



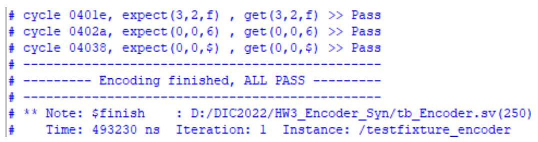
```
# cycle 0401e, expect(3,2,f) , get(3,2,f) >> Pass
# cycle 0402a, expect(0,0,6) , get(0,0,6) >> Pass
# cycle 04038, expect(0,0,$) , get(0,0,$) >> Pass
# -----------------------------------------------
# --------- Encoding finished, ALL PASS ---------
# -----------------------------------------------
# ** Note: $finish    : D:/DIC2022/HW3_Encoder_Syn/tb_Encoder.sv(250)
#    Time: 493230 ns  Iteration: 1  Instance: /testfixture_encoder
```

(your pre-sim result) decoder img1



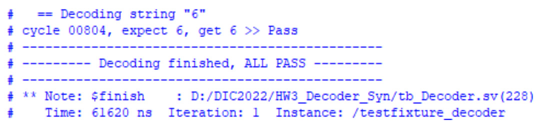
```
#   == Decoding string "6"
# cycle 00804, expect 6, get 6 >> Pass
# -----------------------------------------------
# --------- Decoding finished, ALL PASS ---------
# -----------------------------------------------
# ** Note: $finish    : D:/DIC2022/HW3_Decoder_Sim/tb_Decoder.sv(228)
#    Time: 61620 ns  Iteration: 1  Instance: /testfixture_decoder
```

(your post-sim result) encoder img1



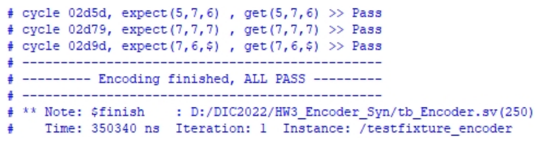
```
# cycle 0401e, expect(3,2,f) , get(3,2,f) >> Pass
# cycle 0402a, expect(0,0,6) , get(0,0,6) >> Pass
# cycle 04038, expect(0,0,$) , get(0,0,$) >> Pass
# -----------------------------------------------
# --------- Encoding finished, ALL PASS ---------
# -----------------------------------------------
# ** Note: $finish    : D:/DIC2022/HW3_Encoder_Syn/tb_Encoder.sv(250)
#    Time: 493230 ns  Iteration: 1  Instance: /testfixture_encoder
```

(your post-sim result) decoder img1



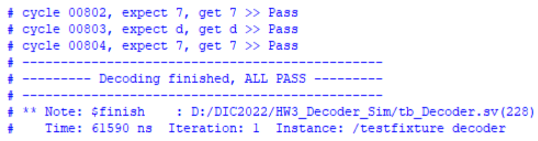
```
#   == Decoding string "6"
# cycle 00804, expect 6, get 6 >> Pass
# -----------------------------------------------
# --------- Decoding finished, ALL PASS ---------
# -----------------------------------------------
# ** Note: $finish    : D:/DIC2022/HW3_Decoder_Syn/tb_Decoder.sv(228)
#    Time: 61620 ns  Iteration: 1  Instance: /testfixture_decoder
```

(your pre-sim result) encoder img2



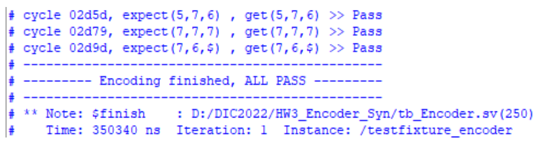
```
# cycle 02d5d, expect(5,7,6) , get(5,7,6) >> Pass
# cycle 02d79, expect(7,7,7) , get(7,7,7) >> Pass
# cycle 02d9d, expect(7,6,$) , get(7,6,$) >> Pass
# -----------------------------------------------
# --------- Encoding finished, ALL PASS ---------
# -----------------------------------------------
# ** Note: $finish    : D:/DIC2022/HW3_Encoder_Syn/tb_Encoder.sv(250)
#    Time: 350340 ns  Iteration: 1  Instance: /testfixture_encoder
```

(your pre-sim result) decoder img2



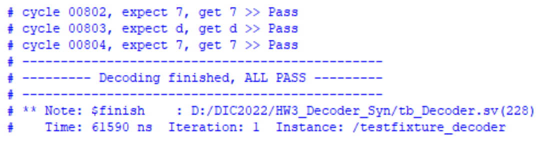
```
# cycle 00802, expect 7, get 7 >> Pass
# cycle 00803, expect d, get d >> Pass
# cycle 00804, expect 7, get 7 >> Pass
# -----------------------------------------------
# --------- Decoding finished, ALL PASS ---------
# -----------------------------------------------
# ** Note: $finish    : D:/DIC2022/HW3_Decoder_Sim/tb_Decoder.sv(228)
#    Time: 61590 ns  Iteration: 1  Instance: /testfixture decoder
```

(your post-sim result) encoder img2



```
# cycle 02d5d, expect(5,7,6) , get(5,7,6) >> Pass
# cycle 02d79, expect(7,7,7) , get(7,7,7) >> Pass
# cycle 02d9d, expect(7,6,$) , get(7,6,$) >> Pass
# -----------------------------------------------
# --------- Encoding finished, ALL PASS ---------
# -----------------------------------------------
# ** Note: $finish    : D:/DIC2022/HW3_Encoder_Syn/tb_Encoder.sv(250)
#    Time: 350340 ns  Iteration: 1  Instance: /testfixture_encoder
```

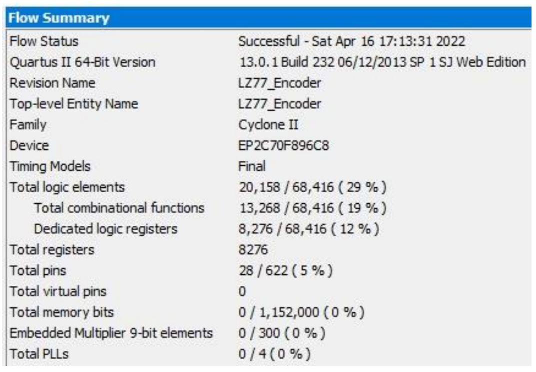
(your post-sim result) decoder img2



```
# cycle 00802, expect 7, get 7 >> Pass
# cycle 00803, expect d, get d >> Pass
# cycle 00804, expect 7, get 7 >> Pass
# -----------------------------------------------
# --------- Decoding finished, ALL PASS ---------
# -----------------------------------------------
# ** Note: $finish    : D:/DIC2022/HW3_Decoder_Syn/tb_Decoder.sv(228)
#    Time: 61590 ns  Iteration: 1  Instance: /testfixture_decoder
```

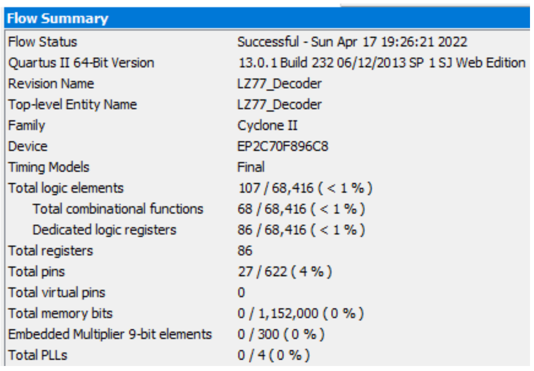
Synthesis Result	**encoder**	**decoder**
Total logic elements	20158	107
Total memory bit	0	0
Embedded multiplier 9-bit element	0	0
Simulation time img0	509190 (ns)	61590 (ns)
Simulation time img1	493230 (ns)	61620 (ns)
Simulation time img2	350340 (ns)	61590 (ns)

(your flow summary) encoder



Flow Summary	
Flow Status	Successful - Sat Apr 16 17:13:31 2022
Quartus II 64-Bit Version	13.0.1 Build 232 06/12/2013 SP 1 SJ Web Edition
Revision Name	LZ77_Encoder
Top-level Entity Name	LZ77_Encoder
Family	Cyclone II
Device	EP2C70F896C8
Timing Models	Final
Total logic elements	20,158 / 68,416 (29 %)
Total combinational functions	13,268 / 68,416 (19 %)
Dedicated logic registers	8,276 / 68,416 (12 %)
Total registers	8276
Total pins	28 / 622 (5 %)
Total virtual pins	0
Total memory bits	0 / 1,152,000 (0 %)
Embedded Multiplier 9-bit elements	0 / 300 (0 %)
Total PLLs	0 / 4 (0 %)

(your flow summary) decoder



Flow Summary	
Flow Status	Successful - Sun Apr 17 19:26:21 2022
Quartus II 64-Bit Version	13.0.1 Build 232 06/12/2013 SP 1 SJ Web Edition
Revision Name	LZ77_Decoder
Top-level Entity Name	LZ77_Decoder
Family	Cyclone II
Device	EP2C70F896C8
Timing Models	Final
Total logic elements	107 / 68,416 (< 1 %)
Total combinational functions	68 / 68,416 (< 1 %)
Dedicated logic registers	86 / 68,416 (< 1 %)
Total registers	86
Total pins	27 / 622 (4 %)
Total virtual pins	0
Total memory bits	0 / 1,152,000 (0 %)
Embedded Multiplier 9-bit elements	0 / 300 (0 %)
Total PLLs	0 / 4 (0 %)

Description of your design

在這次的作業中，有不小心使用到不能合成的電路，在非 clock 的訊號線上，使用了 posedge 的判斷，這樣的做法雖然在跑模擬的時候，可以成功完成，但在跑合成的時候會出現 error，但因為不知道要怎麼不使用到 posedge 來做判斷，跟同學討論他建議我可以把外面的電路拉進 output logic 裡，也就是把 datapath 和 output logic 寫在一起，不過這樣的做法，output logic 將變成一個循序電路，整個電路的面積會比較大，如果要優化這個電路的面積的話，應該可以像助教說的一樣，把 Datapath 拉出來，用組合電路去取代掉不必要的循序電路，應該會是一個比較好的做法。

這次的作業可以分為兩個部分：
第一個部分為 Encoder，可以根據做的事情分成四個狀態，分別是輸入、編碼、輸出以及完成這四個狀態。首先是 datain_state 在 reset 訊號被放下後，資料開始輸入給這個電路的記憶體 data_mem，這個 data_mem 有 2050 個 4 位元的暫存器，每個 clk 都存放一個輸入，再輸入達到 2050，也就是等同於圖片的大小時，會進入下一個狀態，準備開始編碼。接著是 encode_state，在這個狀態下，要先找看看 search_buffer 裡面有沒有與 look_ahead_buffer 裡的值相等的情況，有相等的情況下，我們要找匹配最長的情況，並進到下一個狀態；或是如果都沒有相等的值，在達到 search_buffer 設定的寬度時，也要進到下一個狀態。在 result_state，我們將根據在 encode_state 得到的各個數據，去輸出各個階段編碼的結果，並進到 finish_state。最後的 finish_state 則是要調整 search_buffer_address 的初始位置，並把 search_count、search_start 與 look_ahead_count 歸零，以重新進行下一次的編碼。
第二個部分為 Decoder，Decode 則不分狀態，做的是 encoder 的反運算。

*Scoring = (Total logic elements + total memory bit + 9*embedded multiplier 9-bit element)*

*Scoring of Encoder = 20158 + 0 + 9*0 = 20158*

*Scoring of Decoder = 107+ 0 + 9*0 = 107*