Final Project Report

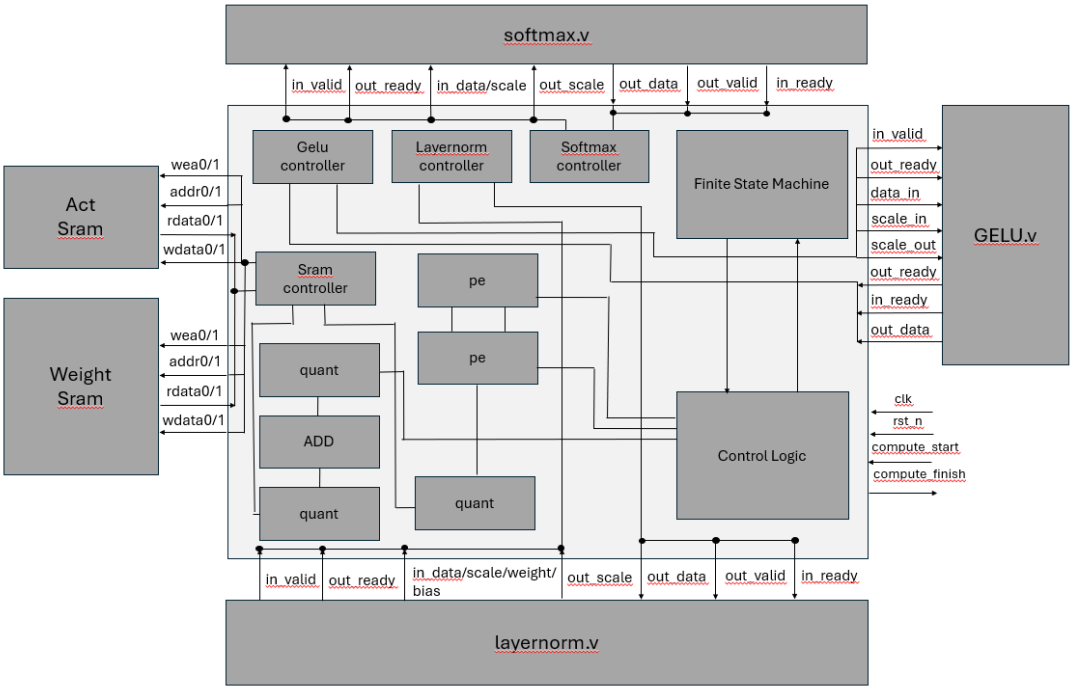
Student ID: 112061533 Name: 潘金生

1. Design concept (You may write the report in Chinese):
   - Please provide block diagrams of the overall hardware architecture and each component. Please explain the purpose and the way they work together.

     這次主要分成三個 module 來撰寫，其中分為 top module bert_encoder 和 submodule pe and quant 去實作，透過 top module 的 Finite State Machine 去控制 pe 和 quant 的操作，除此之外，在 bert_encoder module 裡寫了很多控制訊號去完成讀寫 sram data 的操作。

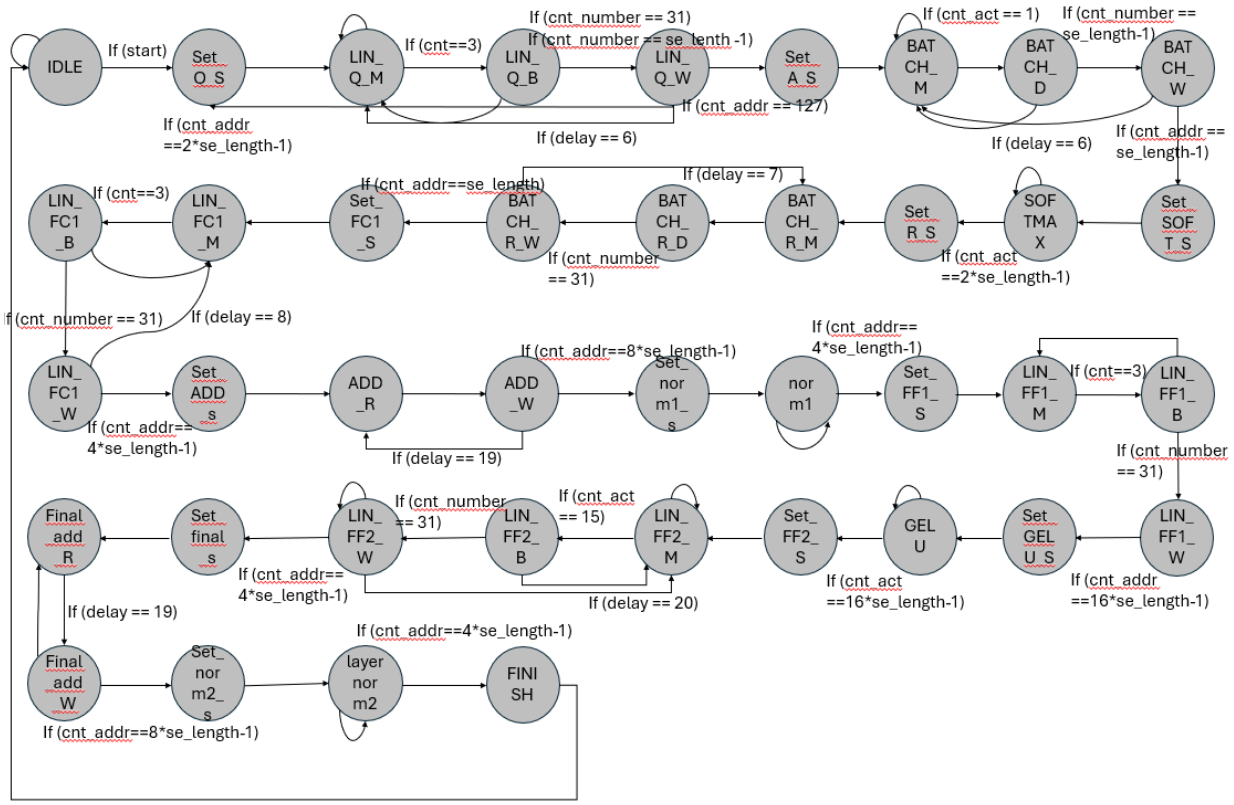
     在本次作業中，1 個 pe 為 16 個 MAC，本次採用兩個 pe 為 32 個 MAC 去實現 bert_encoder 的運算。除此之外，可以看到額外的 quant 和 add，主要用來在流程中的加法位置，其目的可以切 cycle，讓他的 path 可以不要那麼長。值得注意的是，在針對 sram 讀寫時，在轉置的過程中，不是像流程圖的過程，而是擺放的時候直接考慮轉置位置擺放進去的，剩餘的就是在特定的 state 跟 non-linear model 進行 handshake。

     a. Bert_encoder: 用來控制 address 和 wea 參數，判斷何時需要讀入或寫回輸出，內部為一個很大的 FSM 來控制目前的狀態和做什麼事。
     b. Quant: 用來乘上 scale 和進行 shift 16 bits 且 clamp 大小為(-128,127)中間範圍。
     c. Add: 加法的動作，中間有切 1 個 cycle。
     d. Pe: 乘加運算器，用來執行 weight 和 activation 的乘加運算，中間有切一個 cycle。
     e. Controller: 這些 controller 都是寫在 top module 裡的，其目的都是為了進行 handshake 和讀寫記憶體的動作。



- Please provide state diagram of your FSM (if any), and its detailed description. If no FSM is used in your design, please explain your control flow.

這次的 state 分得有點太多，基本上就是按照 spec 提供的流程圖一步一步慢慢做，理論上有些 state 是可以共用的，但我把它分開了，下面有每個 state 在做什麼更詳細的介紹。



IDLE: 等待 Compute_start，Compute_start 訊號為 1 即開始執行。
Set_Q_S: 設置 query, key 和 value 的 scale。
LIN_Q_M: 進行 query, key 和 value 的轉換，乘上權重。
LIN_Q_B: 設置 query, key 和 value 的 bias，以便後續線性轉換相加
LIN_Q_W: 寫入 sram 暫存，且考慮轉置後擺放的位置。
Set_A_S: 設置 attention_score scale
BATCH_M: 進行 Batch 乘法資料所需地址
BATCH_D: 讀取資料延遲 state
BATCH_W: 寫入 Sram 暫存
Set_SOFT_S: 設置 softmax 需要的 scale
SOFTMAX: 進行 softmax handshake 並寫入 sram
Set_R_S: 設置 attention_result scale
BATCH_R_M: 進行 Batch 乘法資料所需地址
BATCH_R_D: 讀取資料延遲 state
BATCH_R_W: 寫入 Sram 暫存
Set_FC1_S: 設置 attention_output scale
LIN_FC1_M: 進行 FC1 乘法
LIN_FC1_B: 設定 FC1 bias

LIN_FC1_W: 寫入 sram 暫存
Set_ADD_S: 設定 attention_residual / attention_add scale
ADD_R: 讀取需要 add 的資料
ADD_W: 寫入 sram 暫存
Set_norm1_s: 設置 layernorm 相關的 scale
Norm1: 進行 layernorm handshake 並寫入 sram
Set_FF1_S: 設定 FF1 scale
LIN_FF1_M: 進行 FF1 乘法
LIN_FF1_B: 設置 FF1 bias，並相加
LIN_FF1_W: 寫入 sram 暫存
Set_GELU_S: 設置 gelu 相關的 scale
GELU: 進行 gelu handshake 並寫入 sram
Set_FF2_S: 設定 FF2 scale
LIN_FF2_M: 進行 FF2 乘法
LIN_FF2_B: 設置 FF2 bias，並相加
LIN_FF2_W: 寫入 sram 暫存
Set_final_s: 設定 FFN_add scale
Final_add_r: 讀取需要 add 的資料
Final_add_w: 寫入 sram 暫存
Set_norm2_s: 設置 layernorm 相關的 scale
Norm2: 進行 layernorm handshake 並寫入 sram
Finish: 設定 compute_finish = 1，代表完成

- Please explain your dataflow in detail.

How many cycle do you need to read/write weights/activations:

Read weights/activations: 在給地址之後，會延遲 1.5 個 clock，由於我們有檔 FF，所以總共會延遲 2 個 CLK
Write weights/activations: 由於我們有擋 FF，需要 1 個 CLK 寫入。
PE: 切 1 個 clock
下面以 sequence length = 28 為例:

state = Set_Q_S, LIN_Q_M, LIN_Q_B, LIN_Q_W:

在進行資料運算時，都會先有一個 clk 負責給 scale 地址，將它存起來以便後續使用。
Query:由於是 ram 讀寫是可以一次使用 2 個 address，我的方式是累積到 2 筆資料在一次寫回去，因此到寫一次需要花費:5 * 32 + 3(資料讀出來需要延遲+pe 切 1 個 clock) + 1(寫回)，故需要 164 個 clock，需要寫回(2*28*64*8)/(128*2) = 112 次，故需要 164*112 = 18368 clock。
Key: 與 Query 相同。
Value: 這裡的存放資料方式比較不同，因為為了後續比較好讀取資料，這次

一次存放的是 sequence length 長度的大小，因此根據 sequence length 的不同，所花費的 cycle 數不同，需要花費 5*28+3+1 共需 144 個 clock，需要寫回 128 次，故需要 144*128 = 18432 clock。

state = Set_A_S, BATCH_M, BATCH_D, BATCH_W:

在進行資料運算時，會給一個 clk 負責給 scale 地址，存起來以便後續使用。
乘法算完後需要有一個延遲時間為了讓 pe 累加歸 0，因此每一次所需花費的 cycle 數為(2+1(pe 歸 0))*28 + 3 + 1，故寫回一次需要 88 個 clock。共需要寫 56 次，因此需要花費 88*56 = 4928 個 clock。

state = Set_SOFT_S, SOFTMAX:

在進行資料運算時，都會先有一個 clk 負責給 scale 地址，將它存起來以便後續使用。
透過 softmax 的 handshake，一次經過 softmax 之後並寫回地址需要 7 個 clock，共需要 56 次，故需要 7*56 = 392 clock。

state = Set_R_S, BATCH_R_M, BATCH_R_D, BATCH_R_W

在進行資料運算時，都會先有一個 clk 負責給 scale 地址，將它存起來以便後續使用。
與先前 batch 乘法類似，需要有一個 clk 讓 pe 歸 0，因此需要寫回兩筆資料所需花費的 cycle 數為 2*64/2 + 3 + 1 = 68 個 clock，需要 112 次，共需花費 7616 個 clock。

state = Set_FC1_S, LIN_FC1_M, LIN_FC1_B, LIN_FC1_W

在進行資料運算時，都會先有一個 clk 負責給 scale 地址，將它存起來以便後續使用。
需要花費 164 個 clock 讓資料讀出相乘和寫入資料(5*32)+3+1，需要寫 112 次，故需要 18368 clock。

state = Set_ADD_S, ADD_R, ADD_W:

在進行資料運算時，都會先有一個 clk 負責給 scale 地址，將它存起來以便後續使用。
這裡是採用將地址每一筆讀出來並進行 quantization 和相加的動作一個地址總共有 16 筆資料，故需要 16+3+1+1，共 21 個 clock，需要 28*128*8/128 次，共 4704 個 clock。

state = Set_norm1_S, norm1:

在進行資料運算時，都會先有一個 clk 負責給 scale 地址，將它存起來以便後續使用。
因為要連傳四筆，加上 weight 和 bias，故需要 8 個 clock 來完成，加上 handshake 以及寫回去，故做完一次需要 18 個 clock，共需要 28 次，故需要 504 個 clock。

state = Set_FF1_S, LIN_FF1_S, LIN_FF1_M, LIN_FF1_W:

在進行資料運算時，都會先有一個 clk 負責給 scale 地址，將它存起來以便後續使用。

與先前一樣，需要花費 164 個 clock 讓資料相乘和寫入，但這次需要 28*512*8/(128*2)，共需要 448 次，故總共需要 73472 個 clock。

state = Set_GELU_S, GELU:

在進行資料運算時，都會先有一個 clk 負責給 scale 地址，將它存起來以便後續使用。

透過 GELU 的 handshake，一次經過 GELU 之後並寫回地址需要 7 個 clock，共需要(28*512*8)/256=448 次，故需要 7*448 = 3136 clock。

state = Set_FF2_S, LIN_FF2_S, LIN_FF2_M, LIN_FF2_W:

在進行資料運算時，都會先有一個 clk 負責給 scale 地址，將它存起來以便後續使用。

需要花 17*32+3+1 個 clock 完成寫回資料，需要花 28*128*8/256 共需要 112 次，總共需要花 61376 個 clock。

state = Set_final_S, Final_add_r, Final_add_w:

在進行資料運算時，都會先有一個 clk 負責給 scale 地址，將它存起來以便後續使用。

需要花 21 個 clock 完成寫回資料，共需要花 224 次，故需要 4704 個 clock。

state = Set_norm2_S, norm2:

在進行資料運算時，都會先有一個 clk 負責給 scale 地址，將它存起來以便後續使用。

需要 18 個 clock 來寫回資料，共需要 28 次，故需要 504 個 clock。

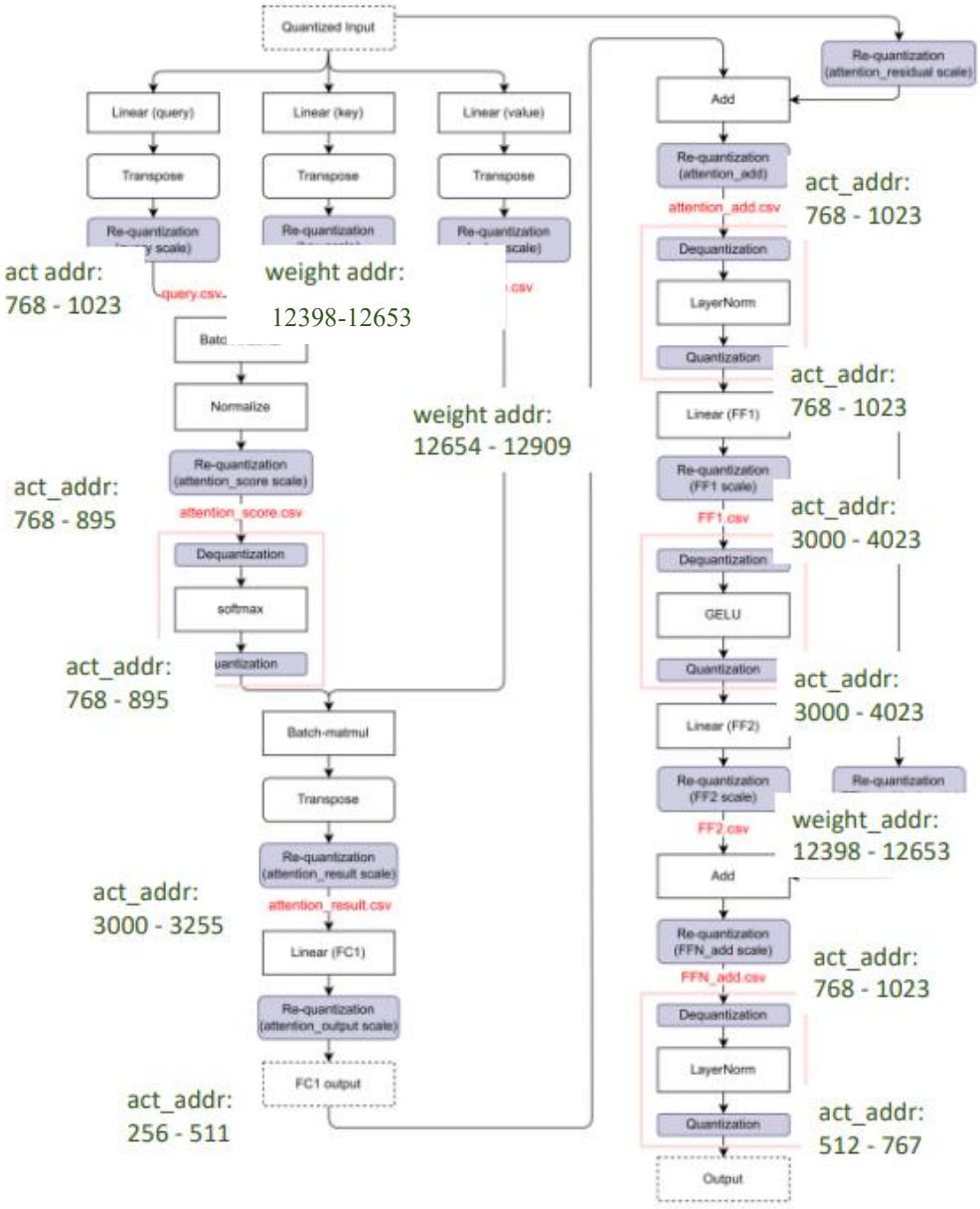
How do you compute the result in how many cycle?

根據上題詳細的介紹，可以看出基本上是按照 spec 上的流程執行，並在過程中有寫回 sram，將上述所需要的 cycle 相加，最後在 sequence length 等於 28 的情況下，所花費的 cycle 數為 234890 個 clock 計算 result。

What data do you keep in register for later reuse?

1. 我會有一個 state，這部分是用來存後續都會用到的 scale。
2. 由於我某些 state 暫存在 sram 的時候，是採用雙 address 進行寫入，也就是一次可以寫入 32 筆資料，這時，每當我有一筆新資料算出時，我會先把他 shift 到 wdata 中，累積到 32 筆時，才寫入。
3. 大部分的 data 都是共用同個 pe 和 quant。

- Please explain the usage of the free-to-use space in the SRAMs.
  附圖為我本次使用 free-to-use 的圖，基本上我只要算完某個地方，我就會先存起來以便後續繼續運算，所以可以看到很多地方都有存，並且存的位置幾乎都是相同的，就不會造成超過 free-to-use 不夠用的情況。附圖為 sequence 最大時，所需要存的位置，確保不會超過 free-to-use range。



- Any additional information you want to provide.
  1. 在討論區中有提到，non-linear 的 IO 不用擋 Flip-Flop，但如果 valid 和 ready，在 code 撰寫的過程中，是有關係的，例如: if(valid) ready =1，這種，在合成的時候，timing Report 就會報錯，會顯示不符合 Slack，因此兩個 IO 是需要獨立的。
  2. 另外在 reset 前我會遇到有 timing violation(hold)的問題，討論區助教有提到可以忽略。

2. Result

本次作業採用 clock period 3ns 去合成跑在 3.1ns 的 gate-level simulation 來避免 setup/hold time 問題(sequence length = 28)

Item	Description	Unit
RTL simulation	PASS	---
Gate-level simulation	PASS	---
Gate-level simulation clock period	3.1	ns
Gate-level simulation latency	234890	cycles
Total cell area	67954.632016	um^2

3. Spyglass Report

由於有些 input port 沒使用到，故會出現 warning，但不影響合成和模擬。

```
############### Non-BuiltIn -> Goal=lint/lint_rtl ###############
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
ID       Rule    Alias    Severity    File                          Line    Wt    Message
================================================================================================
[7]      W240             Warning     ../hdl/bert_encoder.v         35      10    Input 'softmax_data_out_valid' declared but not read.[Hierarchy: ':bert_encoder']
[6]      W240             Warning     ../hdl/bert_encoder.v         36      10    Input 'softmax_data_in_ready' declared but not read.[Hierarchy: ':bert_encoder']
[5]      W240             Warning     ../hdl/bert_encoder.v         49      10    Input 'layernorm_data_out_valid' declared but not read.[Hierarchy: ':bert_encoder']
[4]      W240             Warning     ../hdl/bert_encoder.v         50      10    Input 'layernorm_data_in_ready' declared but not read.[Hierarchy: ':bert_encoder']
[3]      W240             Warning     ../hdl/bert_encoder.v         59      10    Input 'gelu_data_out_valid' declared but not read.[Hierarchy: ':bert_encoder']
[2]      W240             Warning     ../hdl/bert_encoder.v         60      10    Input 'gelu_data_in_ready' declared but not read.[Hierarchy: ':bert_encoder']
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

這部分要加入 SGDC constraint，來讓 Design Compiler 找到 clocks，不過我有去檢查波型，發現我所設計 FF 有正常運作，因此在這裡忽略它。

```
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
MORESIMPLE REPORT:
############### FATAL MESSAGES ###############
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
ID       Rule         Alias    Severity    File                          Line    Wt    Message
================================================================================================
[2]      Av_init01            Fatal       ../hdl/bert_encoder.v         1       10    Could not find clocks for all the flops. Please add clock SGDC constraint to the design
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

軟體廠商授予與我們的 license 限制，可以忽略。

```
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
MORESIMPLE REPORT:
############### FATAL MESSAGES ###############
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
ID       Rule            Alias    Severity    File    Line    Wt    Message
================================================================================================
[0]      Av_license01            FATAL       N.A.    0       2     'Advanced Lint Policy' not run due to unavailability of Auto_Verify license feature
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

4. Others (optional)

- 謝謝助教在討論區的回覆。