113 學年度

國立中山大學

課程名稱：算數處理器與實作

題目：Design of Single-Precision Floating-Point Adders and Multipliers

作業/成果報告/專題

授課教師:蕭勝夫
學生學號/班級/姓名:
B103040021/114 資工/謝鎧駿

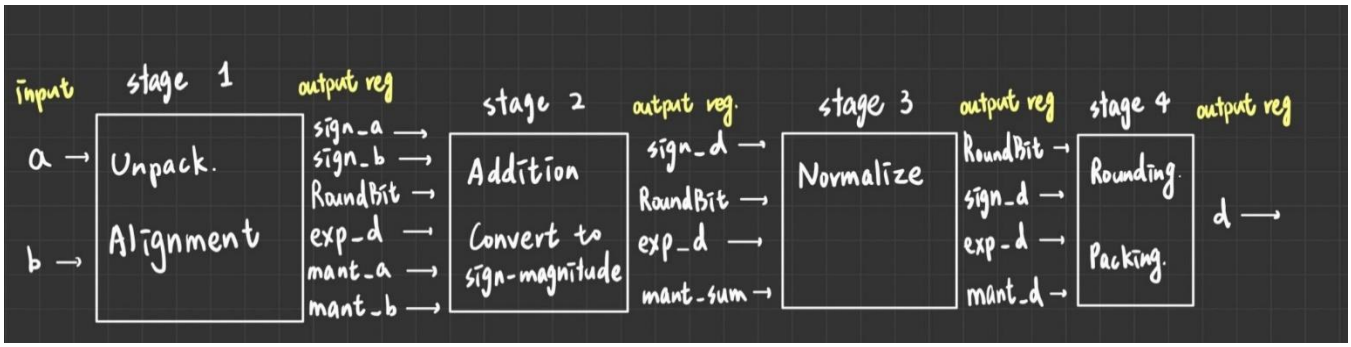
一、數據比較表格：

		Area (um^2)			Delay (ns)	Latency (ns)	Power (W)		
		CL	SL	Total			dynamic	leakage	total
FXP_adder	area	33.125761	0	33.125761	0.874337	0.874337	23.0760	18.3088	41.3848
	mid	50.025601	0	50.025601	0.482147	0.482147	51.5115	29.3665	80.878
	delay	101.917442	0	101.917442	0.089956	0.089956	526.2542	83.2316	609.4858
FLP_adder	area	430.427531	0	430.427531	1.996085	1.996085	173.1295	354.0467	527.1762
	mid	456.710412	0	456.710412	1.248837	1.248837	235.9364	344.7674	580.7038
	delay	922.648338	0	922.648338	0.501589	0.501589	1.1241	841.5132	842.6373
FLP_adder_7	area	459.820810	304.508171	764.328981	0.731806	5.122642	1.2510	542.6097	543.8607
	mid	475.320971	304.508171	779.829142	0.478085	3.346595	1.9054	529.449	531.3544
	delay	593.619853	304.974731	898.594584	0.224364	1.570548	4.0467	634.0710	638.1177
FLP_adder_4	area	440.277130	145.981445	586.258575	0.731806	2.927224	855.042	440.2025	1295.2445
	mid	454.429450	145.981445	600.410896	0.4773305	1.909322	1.2979	429.3313	430.6292
	delay	542.764812	145.981445	688.746257	0.222855	0.89142	2.6113	492.6586	495.2699
FXP_multiplier	area	915.027863	0	915.027863	2.074983	2.074983	536.4115	277.2525	813.664
	mid	983.041944	0	983.041944	1.202865	1.202865	983.9003	529.3001	1513.2004
	delay	1610.305947	0	1610.305947	0.330747	0.330747	7.1930	1.6339	8.8269
FLP_multiplier	area	580.348815	0	580.348815	2.357140	2.357140	274.0397	210.1722	484.2119
	mid	622.183695	0	622.183695	1.37839	1.37839	520.8844	461.9382	982.8226
	delay	947.946256	0	947.946256	0.399639	0.399639	3.1622	941.7064	944.8686
FLP_multiplier_4	area	634.936335	179.055367	813.991702	1.605725	6.4229	488.7879	337.0256	825.8135
	mid	678.326416	179.055367	857.381782	0.994395	3.97758	812.6987	484.5254	1297.2241
	delay	949.760658	179.210886	1128.971544	0.383065	1.53226	2.6201	939.3749	941.995

二、架構圖

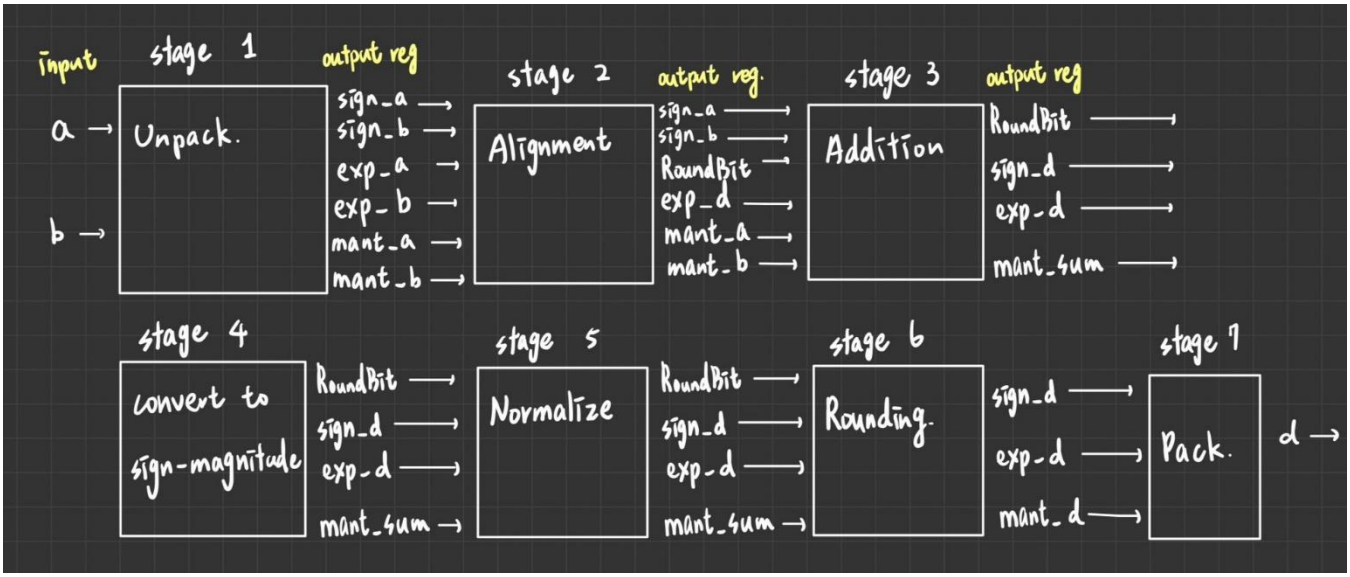
1. FLP_adder_4

   Critical path:stage2



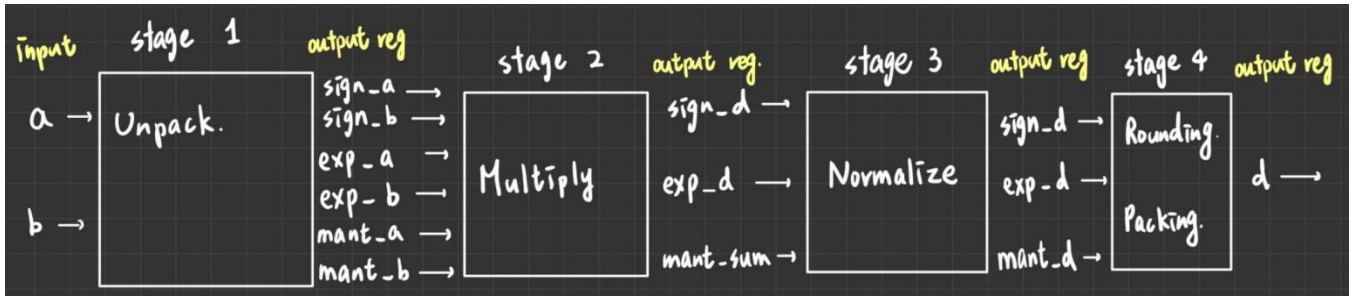
2. FLP_adder_7

   Critical path:stage3



3. FLP_mul_4

   Critical path:stage2



三、Pipeline 數據

(一) Area optimize

1.FLP_adder_4：

Stage1:0.51
Stage2:0.72
Stage3:0.69
Stage4:0.47

2.FLP_adder_7：

Stage1:0.01
Stage2:0.54
Stage3:0.72
Stage4:0.03
Stage5:0.69
Stage6:0.49
Stage7:0.04

3.FLP_mul_4：

Stage1:0.01
Stage2:1.59
Stage3:0.2
Stage4:0.37

(二) Between

1.FLP_adder_4：

Stage1:0.36
Stage2:0.46
Stage3:0.46
Stage4:0.47

2.FLP_adder_7：

Stage1:0.01
Stage2:0.36
Stage3:0.46
Stage4:0.03
Stage5:0.46
Stage6:0.46
Stage7:0.04

3.FLP_mul_4：

Stage1:0.01
Stage2:0.98
Stage3:0.2
Stage4:0.37

(三) Delay optimize

1. FLP_adder_4：
   Stage1:0.21
   Stage2:0.21
   Stage3:0.21
   Stage4:0.21
2. FLP_adder_7：
   Stage1:0.01
   Stage2:0.21
   Stage3:0.21
   Stage4:0.03
   Stage5:0.21
   Stage6:0.19
   Stage7:0.04
3. FLP_mul_4：
   Stage1:0.01
   Stage2:0.37
   Stage3:0.2
   Stage4:0.37

四、心得

這學期同時修了 HDL 跟 ALU，剛好這份作業可以當作 HDL 的進階版，但是第一次撰寫 Pipeline 也是困難重重，一開始是不知道何時該用=何時該用<=，後來才搞懂是計算時一樣是使用=，而計算完要存進 reg 時才是用<=。

處理完這個問題後就成功寫出 Pipeline 的 adder 跟 mul 了，但在進行 logic synthesis 的時候，卻發現 Adder 中的 while loop 居然不能 synthesis，只好改寫成 case 去找 leading 1，最後才成功合成出來。

而整理完所有數據後發現，每個 Pipeline 的 critical path 都是進行加法或乘法的那一個 stage，尤其是乘法器的 case 感覺超花時間的。