國立台灣大學電機資訊學院電子工程學研究所

系統晶片設計實驗
Soc Design Laboratory

Final Project Report

Final Project

學生： M11202109 蘇柏丞
M11202103 陳泓宇
M11202207 呂彥霖

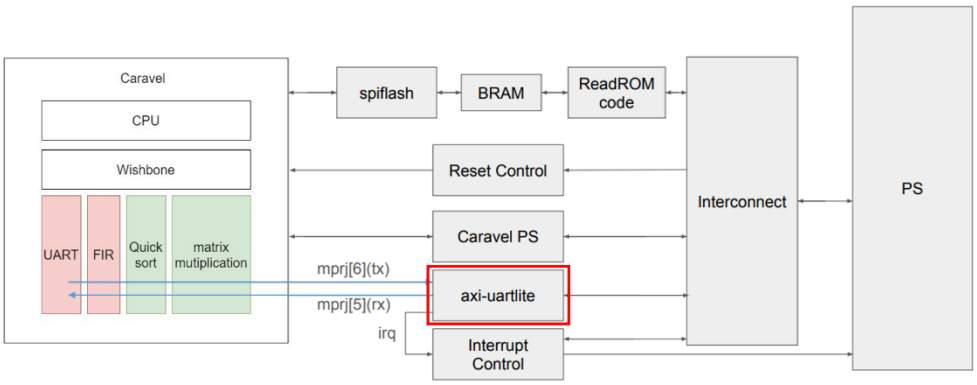
老師： 賴 瑾

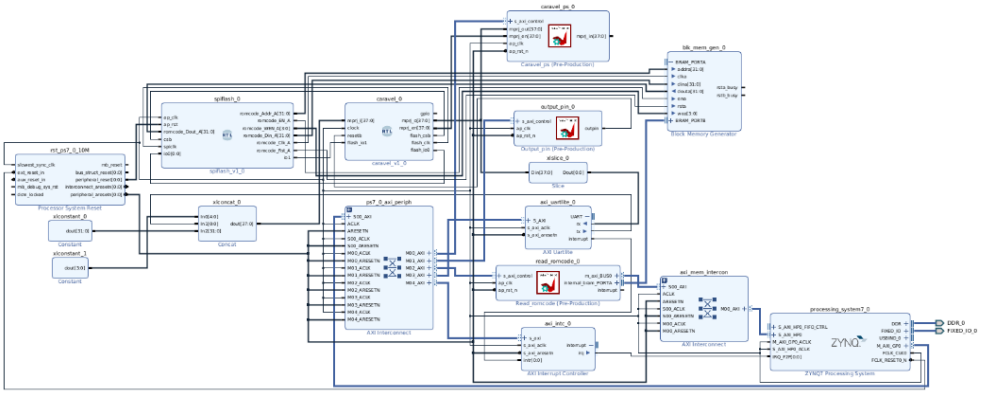
中華民國 112 年 1 月 3 日

一、 Function：

1. An accelerator for matrix multiplication (4 x 4) and quicksort (11) has been designed and integrated into the SoC.
2. The baud rate for the UART has been changed from 9600 to 115200.
3. The type of memory is BRAM with prefetch controller.
4. The designs have be implemented on SoC.
5. The SoC have be programed into FPGA and verified on jupyter notebook

二、 Block Diagram：





三、 Synthesis report：

1. Timing Report

- Synthesis

Setup		Hold		Pulse Width	
Worst Negative Slack (WNS):	8.245 ns	Worst Hold Slack (WHS):	-0.762 ns	Worst Pulse Width Slack (WPWS):	11.250 ns
Total Negative Slack (TNS):	0.000 ns	Total Hold Slack (THS):	-1.523 ns	Total Pulse Width Negative Slack (TPWS):	0.000 ns
Number of Failing Endpoints:	0	Number of Failing Endpoints:	2	Number of Failing Endpoints:	0
Total Number of Endpoints:	17323	Total Number of Endpoints:	17323	Total Number of Endpoints:	6938

Timing constraints are not met.

- Implementation

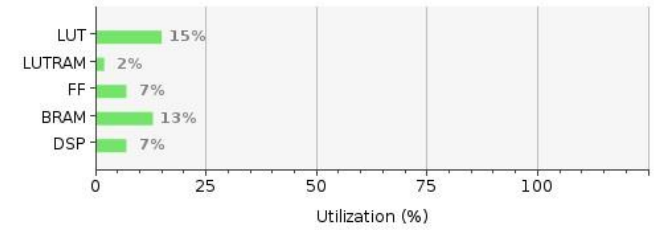
Setup		Hold		Pulse Width	
Worst Negative Slack (WNS):	5.175 ns	Worst Hold Slack (WHS):	0.020 ns	Worst Pulse Width Slack (WPWS):	11.250 ns
Total Negative Slack (TNS):	0.000 ns	Total Hold Slack (THS):	0.000 ns	Total Pulse Width Negative Slack (TPWS):	0.000 ns
Number of Failing Endpoints:	0	Number of Failing Endpoints:	0	Number of Failing Endpoints:	0
Total Number of Endpoints:	16207	Total Number of Endpoints:	16207	Total Number of Endpoints:	6508

All user specified timing constraints are met.

2. Resource Report

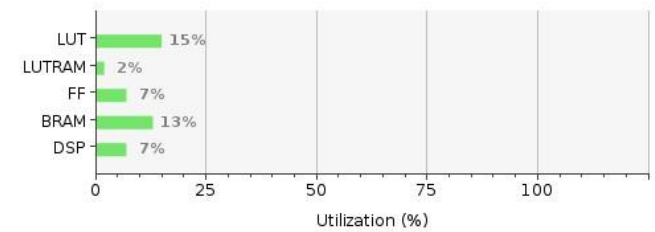
- Synthesis

Resource	Utilization	Available	Utilization %
LUT	7988	53200	15.02
LUTRAM	373	17400	2.14
FF	7799	106400	7.33
BRAM	18	140	12.86
DSP	15	220	6.82



- Implementation

Resource	Utilization	Available	Utilization %
LUT	7988	53200	15.02
LUTRAM	373	17400	2.14
FF	7799	106400	7.33
BRAM	18	140	12.86
DSP	15	220	6.82



四、 Register specification：

1. Base Address Map

User Project	Base Address Map
Memory Starting	3800_0000
FIR Base Address	3600_0000
Matrix Multiplication Base Address	3400_0000
Quick Sort Base Address	3200_0000
UART Base Address	3000_0000

2. Register Address Map – FIR

FIR		
0x00	[0]	ap_start (r/w)
	[1]	ap_done (ro)
	[2]	ap_idle (ro)
	[3]	X[n]_ready to accept input (ro)
	[4]	Y[n] is ready to read
0x10-13	data-length	
0x40-7F	Tap parameters	
0x80-83	X[n] input (r/w)	
0x84-87	Y[n] output (ro)	

3. Register Address Map – Matrix Multiplication

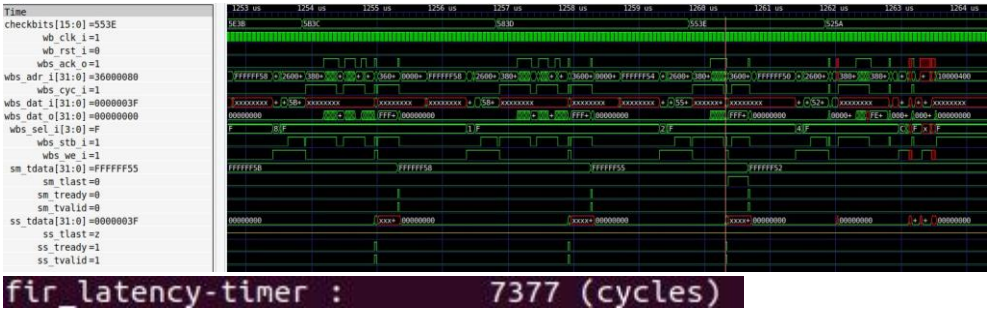
Matrix Multiplication		
0x00	[0]	ap_start (r/w)
	[1]	ap_done (ro)
	[2]	ap_idle (ro)
	[3]	X[n]_ready to accept input (ro)
	[4]	Y[n] is ready to read
0x80-83	X[n] input (r/w)	
0x84-87	Y[n] output (ro)	

4. Register Address Map – Quick Sort

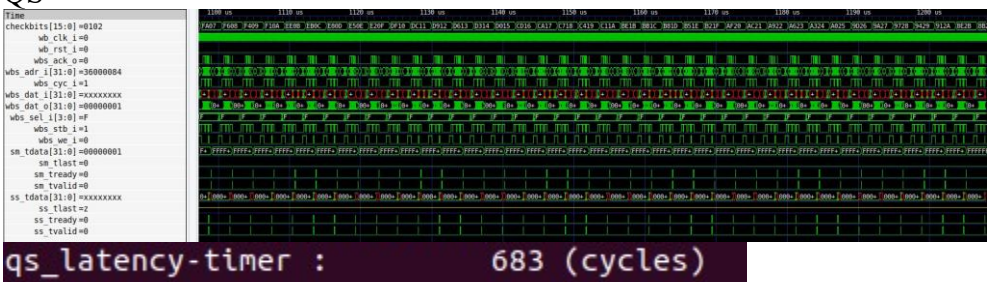
Quick Sort		
0x00	[0]	ap_start (r/w)
	[1]	ap_done (ro)
	[2]	ap_idle (ro)
	[3]	X[n]_ready to accept input (ro)
	[4]	Y[n] is ready to read
0x80-83	X[n] input (r/w)	
0x84-87	Y[n] output (ro)	

五、 Waveform & latency

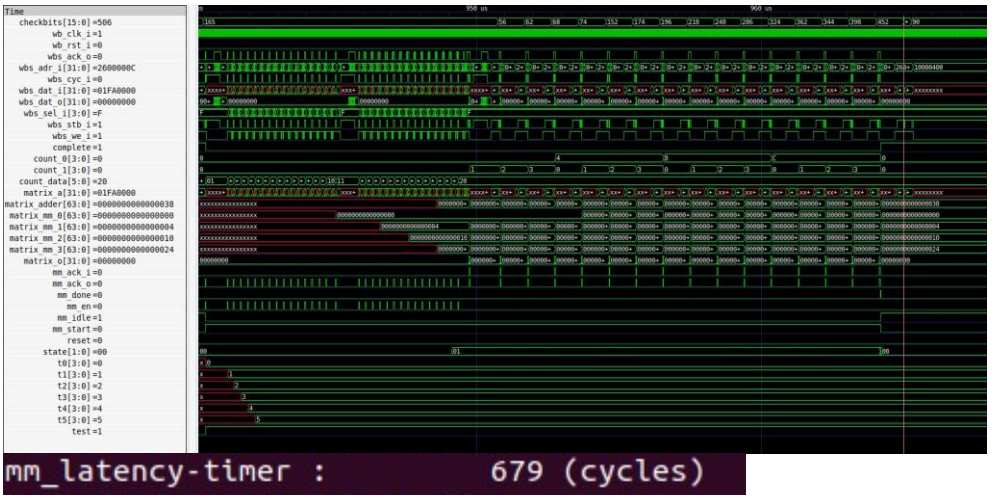
1. FIR



2. QS



3. MM



4. UART

```
Last latency-timer :        91756 (cycles)
```

六、 How to verify answer from notebook：

```
ipOUTPIN.write(0x10, 0)
print("Start Caravel Soc")
ipOUTPIN.write(0x10, 1)
print ("0x1c = ", hex(ipPS.read(0x1c)))
```

```
Start Caravel Soc
0x1c =  0x40
```

```
import time

# Reset FIFOs, enable interrupts
ipUart.write(CTRL_REG, 1<<RST_TX | 1<<RST_RX | 1<<INTR_EN)
print("Waitting for interrupt")

tx_str = ""
for i in range(512):
    tx_str += 'h'

uart_c_start_time = time.time_ns()
ipUart.write(TX_FIFO, ord(tx_str[0]))
i = 1

temp = ""
while(i<len(tx_str)):
#     await intUart.wait()
    buf = ""
    # Read FIFO until valid bit is clear
    while ((ipUart.read(STAT_REG) & (1<<RX_VALID))):
        buf += chr(ipUart.read(RX_FIFO))
        if i<len(tx_str):
            ipUart.write(TX_FIFO, ord(tx_str[i]))
            i=i+1
    temp += buf

uart_c_end_time = time.time_ns()

print("uart loop back latnecy  for ",temp," = ",uart_c_end_time - uart_c_start_time ,"ns")
```

```
Waitting for interrupt
uart loop back latnecy  for  hhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhh
hhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhh
hhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhh
hhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhh
hhhhhhhhhhhhhhhhhhhhhhhhhhhhhhh  =  1163962099 ns
```

```
print ("0x1c = ", hex(ipPS.read(0x1c)))
```

```
0x1c =  0xab610040
```

七、 QoR

- QoR Metric for Computation (matmul, qsort fir) – Based on Simulation
  MM：679 (cycles)
  QS：683 (cycles)
  FIR：7377 (cycles)

  Total cycles：8739 (cycles)

- QoR Metric for Communication (UART) – Based on FPGA, PS
  Latency：1163962099 ns
  QoR Metrics = 1.163962099 – 500 * (1/115200) = 1.1545383

八、 Work Partition：

蘇柏丞	Quick Sort
	PPT
	Implementation
呂彥霖	Matrix Multiplication
	Report
	PS side verification
陳泓宇	UART
	Prefetch controller
	Integration