CS422 Assignment-3

Pipelining the MIPS integer processor

Kundan Kumar 220568
Mahaarajan J 220600
Harshit Srivastava 220444

25th Mar 2025

Part A

Test Case	Instructions	Loads (%)	Stores (%)	Conditional Branches (%)
asm-sim	158	39.24	34.81	0.00
c-sim	1928	17.43	8.20	19.29
endian	1965	19.13	11.04	18.02
factorial	2530	16.48	8.62	18.42
fib	44903	13.59	17.78	8.34
hello	1744	19.09	8.60	19.61
host	14894	20.10	9.92	19.22
ifactorial	2420	16.49	7.85	19.26
ifib	7581	16.99	10.01	17.50
log2	2096	17.89	8.35	19.61
msort	17219	10.59	5.44	18.13
rfib	25668	14.40	17.03	9.51
subreg	1702	19.03	9.40	18.45
towers	82763	20.06	8.49	20.54
vadd	7259	6.14	16.35	10.58

Part B

2.1 Results

Design abbreviations:

- **Design-1**: Complete interlock logic, no bypass, two delay slots
- **Design-2**: One delay slot no bypasses
- **Bypass-1**: EX-EX bypass enabled
- **Final Design**: All bypass paths (EX-EX and MEM-EX) enabled, one delay slot

Test Case	Design-1	Design-2	Bypass-1	Final Design	Load-delay stalls (%)
asm-sim	1.65	1.49	1.35	1.29	0
c-sim	1.84	1.68	1.17	1.01	0
endian	1.77	1.61	1.18	1.01	0
factorial	1.82	1.66	1.19	1.01	0
fib	1.66	1.47	1.11	1.00	0
hello	1.81	1.58	1.14	1.01	0
host	1.67	1.51	1.16	1.01	0
ifactorial	1.80	1.65	1.09	1.01	0
ifib	1.70	1.57	1.14	1.00	0
log2	1.76	1.61	1.15	1.01	0
msort	1.88	1.75	1.16	1.00	0
rfib	1.67	1.45	1.09	1.00	0
subreg	1.75	1.61	1.12	1.02	0
towers	1.58	1.42	1.11	1.00	0
vadd	1.96	1.90	1.05	1.00	0

2.2 Approach and Analysis

- **Extra Branch delay slot removal:** Branch target calculated in EX on the positive half cycle; IF uses new PC in same cycle at the negative half
- **Bypass logic:** Buffer states for all logical registers. Source operands read from buffer when available from the required pipeline register which is set at the ID/RF stage. Also there is no need for a MEM-MEM bypass because the WB happens at the positive half cycle and memory read happens at the negative half of the same cycle.
- **Stall handling:** ID/RF pushes nop's after saving the instruction on unavailable source operands or syscalls. The availability of source operands is tracked by comparison with 2 previous instructions' destination operands.

Design and Pipeline Highlights:

- Syscalls stall pipeline until write-back; fetched instruction discarded — 4 stall cycles per syscall so that the pipeline is filled with nop's when the syscall is executed in WB stage.
- High CPI in some cases is due to syscall-induced stalls, as the compiler fills the load delay slot appropriately even though we check it for interlock logic. And as we can see from the statistics load induced stalls are nil.

Test Case	Instructions	Syscalls	Syscall %
asm-sim	49	3	6.12
c-sim	1825	5	0.27
endian	1864	5	0.27
factorial	2407	5	0.21
fib	44779	5	0.01
hello	1638	5	0.31
host	14519	24	0.17
ifactorial	2293	5	0.22
ifib	7453	5	0.07
log2	1995	5	0.25
msort	17103	5	0.03
rfib	25540	5	0.02
subreg	1702	10	0.59
towers	80651	67	0.08
vadd	7129	5	0.07

Hence as we can see there is a close correlation between percentage of syscalls and the CPI for the programs.