Bailey Morgan

Alex Dripchak

Eric Tu

Josh Eckels

Design Philosophy

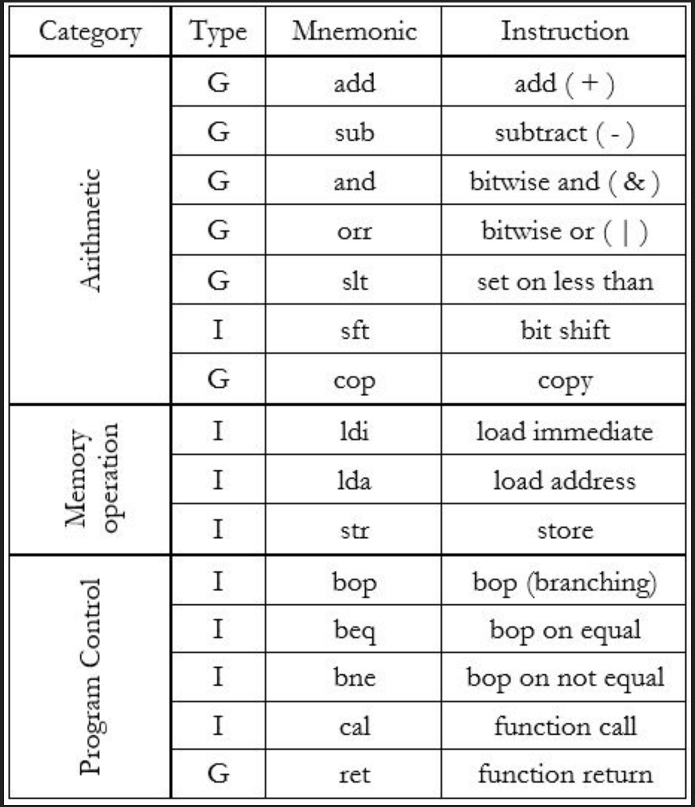
- Design
    - Keep the user in mind
    - Make hardware interfacing simple
- Performance
    - Requires complexity in design
    - Does not require complexity in interface

Architecture

- Load store

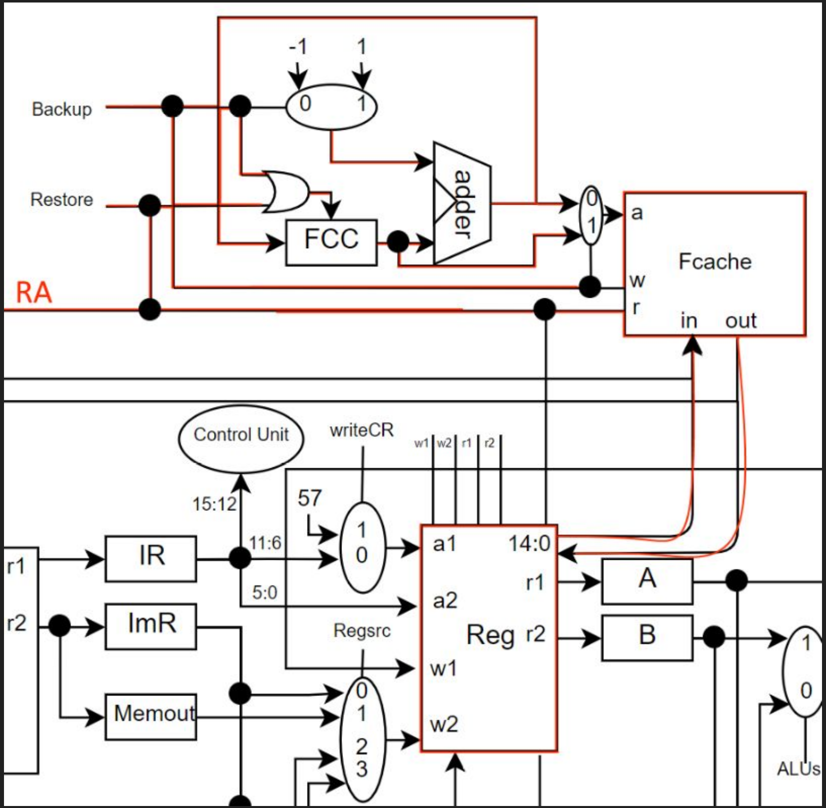
- Accumulator

Instruction Set

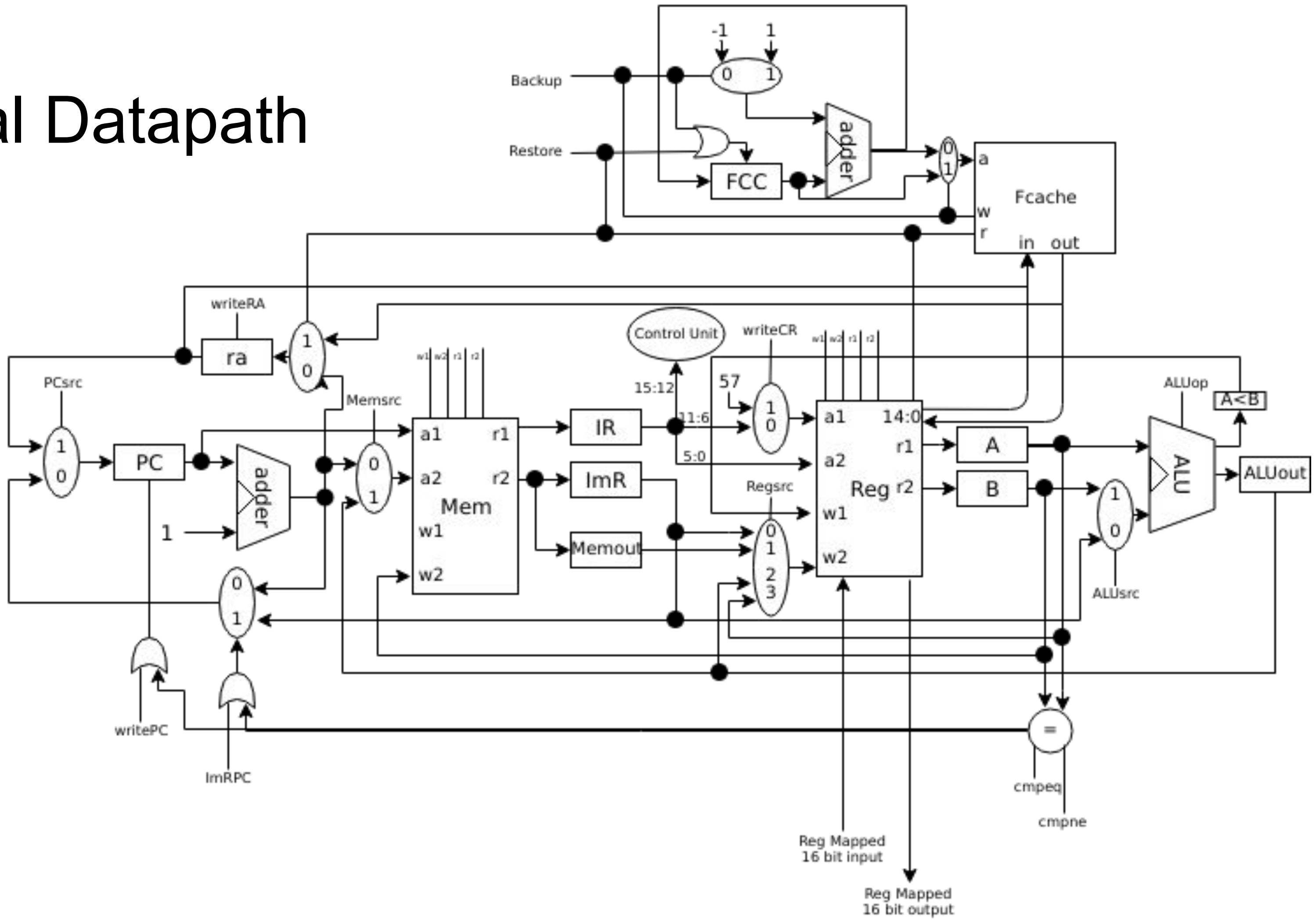


Category	Type	Mnemonic	Instruction
Arithmetic	G	add	add (+)
	G	sub	subtract (-)
	G	and	bitwise and (&)
	G	orr	bitwise or (|)
	G	slt	set on less than
	I	sft	bit shift
	G	cop	copy
Memory operation	I	ldi	load immediate
	I	lda	load address
	I	str	store
Program Control	I	bop	bop (branching)
	I	beq	bop on equal
	I	bne	bop on not equal
	I	cal	function call
	G	ret	function return

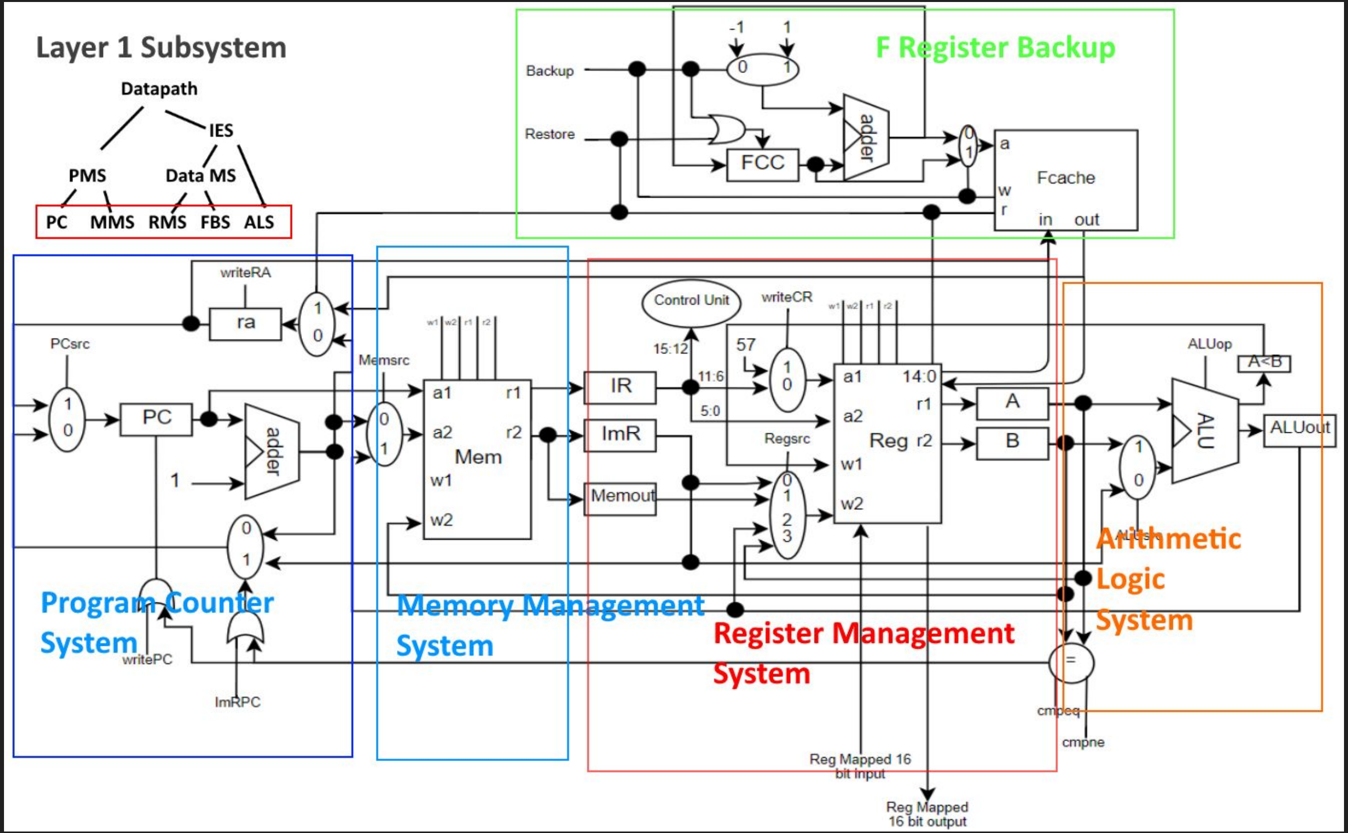
Caching


-1
1
0
1
Backup
Restore
adder
FCC
0
1
a
Fcache
w
r
in
out
RA
Control Unit
writeCR
w1
w2
r1
r2
15:12
57
11:6
5:0
r1
IR
r2
ImR
Memout
Regsrc
0
1
2
3
a1
14:0
r1
a2
Reg
r2
w1
w2
A
B
1
0
ALUs

Final Datapath

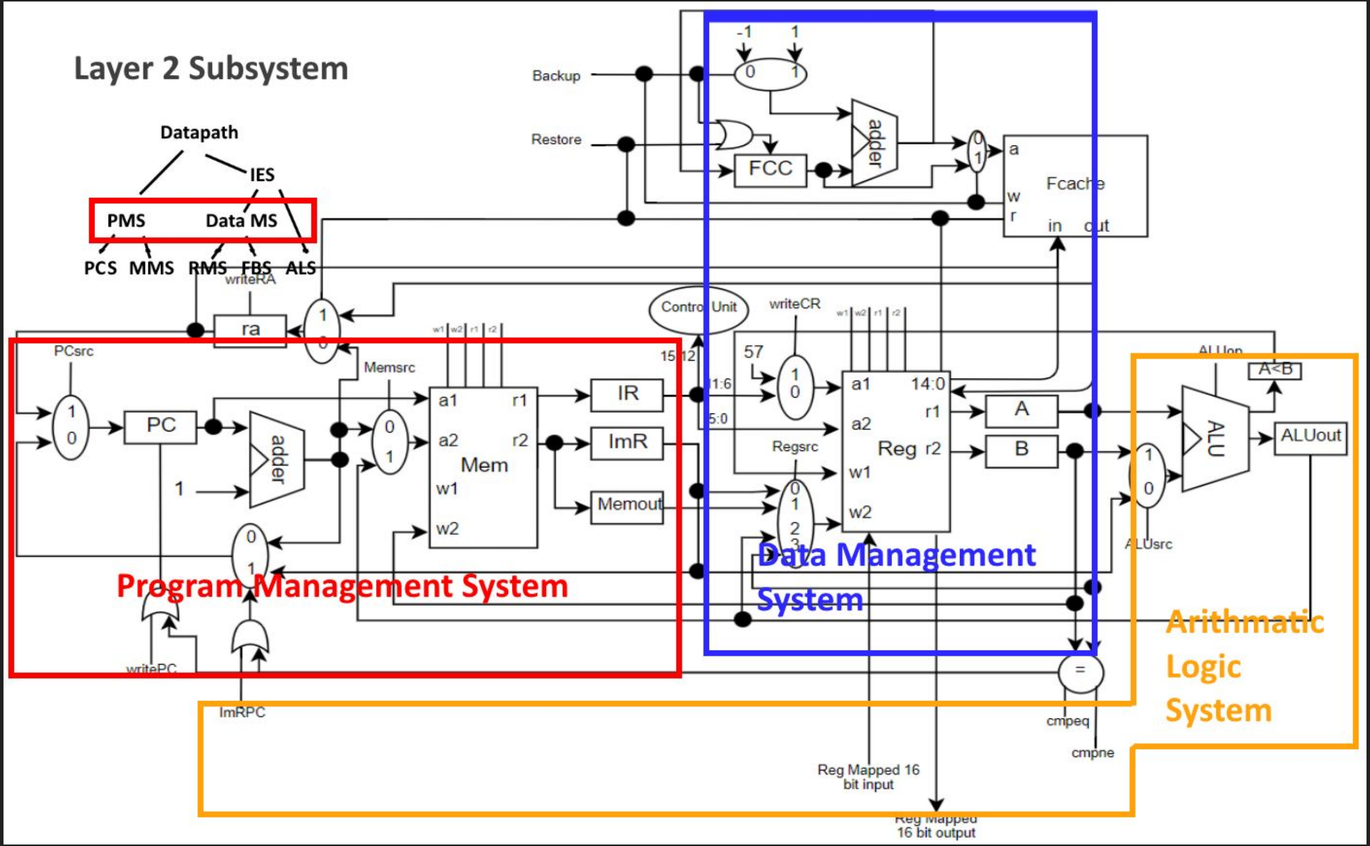


Testing

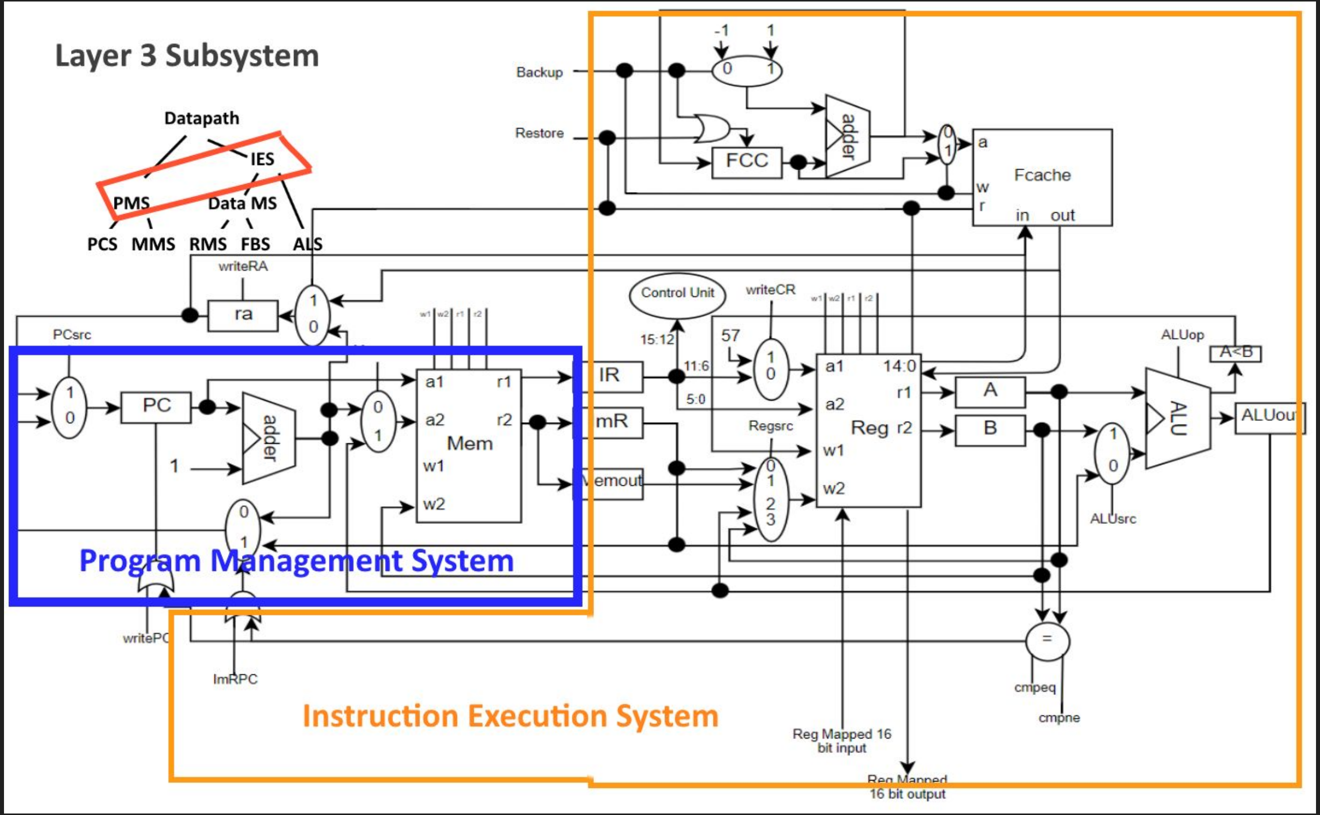


Layer 1 Subsystem
Datapath
IES
PMS
Data MS
PC
MMS
RMS
FBS
ALS
F Register Backup
Backup
Restore
-1
1
0
1
adder
FCC
Fcache
a
w
r
in
out
writeRA
ra
PCsrc
PC
adder
1
Memsrc
Mem
a1
a2
w1
w2
r1
r2
IR
ImR
Memout
Control Unit
writeCR
15:12
11:6
5:0
57
Regsrc
Reg
14:0
A
B
ALUop
A<B
ALU
ALUout
ALUsrc
=
cmpeq
cmpne
Program Counter System
writePC
ImRPC
Memory Management System
Register Management System
Arithmetic Logic System
Reg Mapped 16 bit input
Reg Mapped 16 bit output

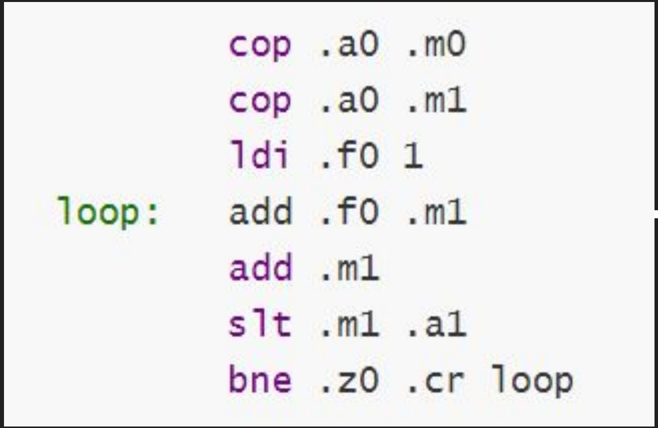
Testing


Layer 2 Subsystem
Datapath
IES
PMS
Data MS
PCS
MMS
RMS
FBS
ALS
Backup
Restore
-1
1
0
1
adder
FCC
Fcache
a
w
r
in
out
writeRA
ra
Control Unit
writeCR
57
PCsrc
PC
adder
Memsrc
Mem
a1
a2
w1
w2
r1
r2
IR
ImR
Memout
15:12
11:6
5:0
Regsrc
Reg
14:0
A
B
ALUop
A<B
ALU
ALUout
ALUsrc
=
cmpeq
cmpne
writePC
ImRPC
Reg Mapped 16 bit input
Reg Mapped 16 bit output
Program Management System
Data Management System
Arithmatic Logic System

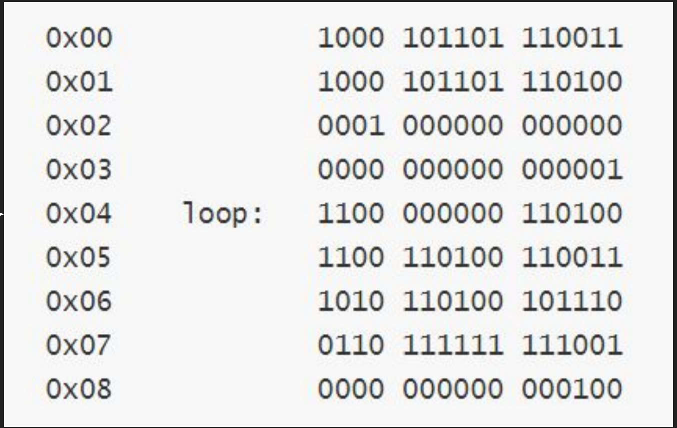
Testing


Layer 3 Subsystem
Datapath
PMS
IES
PCS
MMS
Data MS
RMS
FBS
ALS
Backup
Restore
-1
1
FCC
adder
Fcache
a
w
r
in
out
writeRA
ra
PCsrc
PC
adder
1
Mem
a1
a2
w1
w2
r1
r2
Control Unit
writeCR
15:12
11:6
5:0
57
IR
mR
Memout
Regsrc
Reg
14:0
A
B
ALUop
A<B
ALU
ALUout
ALUsrc
cmpeq
cmpne
writePC
ImRPC
Reg Mapped 16 bit input
Reg Mapped 16 bit output
Program Management System
Instruction Execution System

Assembler



```
        cop .a0 .m0
        cop .a0 .m1
        ldi .f0 1
loop:   add .f0 .m1
        add .m1
        slt .m1 .a1
        bne .z0 .cr loop
```

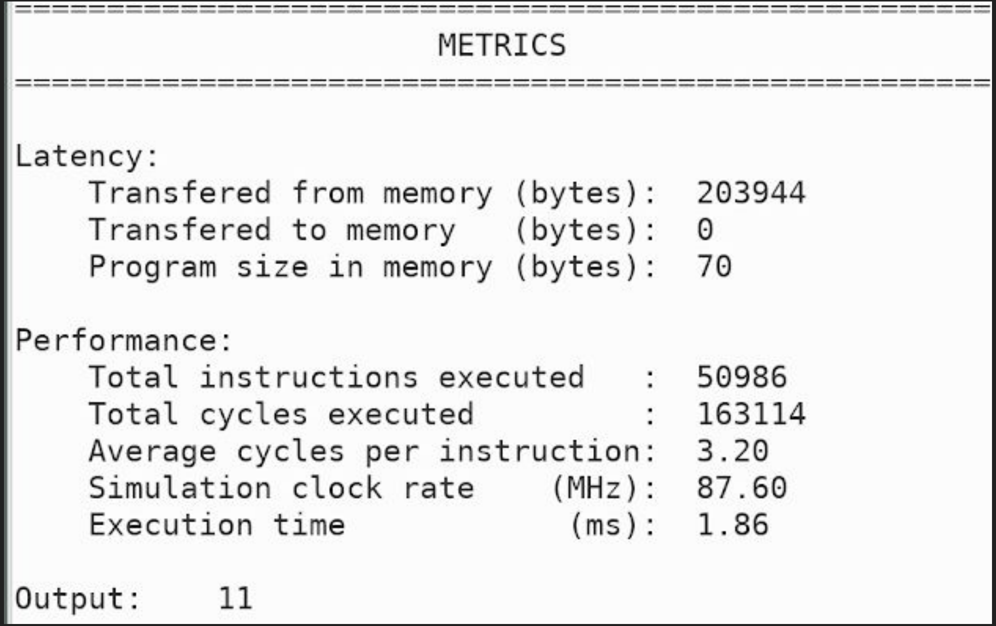


```
0x00          1000 101101 110011
0x01          1000 101101 110100
0x02          0001 000000 000000
0x03          0000 000000 000001
0x04   loop:  1100 000000 110100
0x05          1100 110100 110011
0x06          1010 110100 101110
0x07          0110 111111 111001
0x08          0000 000000 000100
```

Simulator

Performance

Running the relative prime algorithm implemented in our language with 5040 in the simulator gave the following performance metrics.



```
=====================================================
                        METRICS
=====================================================

Latency:
    Transfered from memory (bytes):  203944
    Transfered to memory   (bytes):  0
    Program size in memory (bytes):  70

Performance:
    Total instructions executed   :  50986
    Total cycles executed         :  163114
    Average cycles per instruction:  3.20
    Simulation clock rate    (MHz):  87.60
    Execution time            (ms):  1.86

Output:    11
```

Challenges and Trade-offs

```
[illegible]
[illegible]$ [illegible] 5000
+++++++++++++++++++++++++++++++++++++++++++++++++++++
                       METRICS
=====================================================

Latency:
    Transferred from memory (bytes):  [illegible]
    Transferred to memory   (bytes):  0
    Program size in memory  (bytes):  [illegible]

Performance:
    Total instructions executed    :  [illegible]
    Total cycles executed          :  [illegible]
    Average cycles per instruction :  [illegible]
    Simulation clock rate    (MHz) :  [illegible]
    Execution time            (ms) :  [illegible]

Output:    11

[illegible]$ cp [illegible]
```