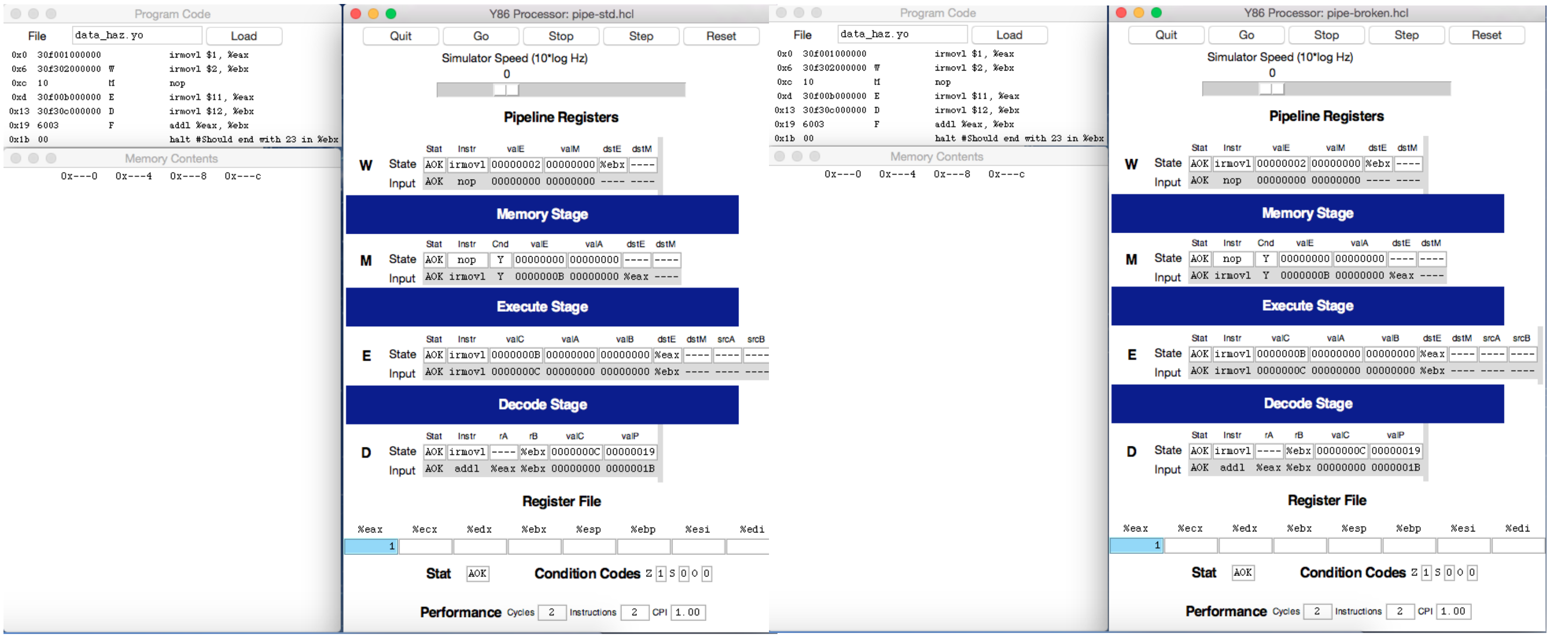
Data Hazard Demonstration

Carter J. Bastian [*]

May 2015



Program Code

File data_haz.yo Load

```
0x0  30f001000000              irmovl $1, %eax
0x6  30f302000000  W           irmovl $2, %ebx
0xc  10            M           nop
0xd  30f00b000000  E           irmovl $11, %eax
0x13 30f30c000000  D           irmovl $12, %ebx
0x19 6003          F           addl %eax, %ebx
0x1b 00                        halt #Should end with 23 in %ebx
```

Memory Contents

```
0x---0   0x---4   0x---8   0x---c
```

Y86 Processor: pipe-std.hcl

Quit Go Stop Step Reset

Simulator Speed (10*log Hz) 0

Pipeline Registers

W	Stat	Instr	valE	valM	dstE	dstM
State	AOK	irmovl	00000002	00000000	%ebx	----
Input	AOK	nop	00000000	00000000	----	----

Memory Stage

M	Stat	Instr	Cnd	valE	valA	dstE	dstM
State	AOK	nop	Y	00000000	00000000	----	----
Input	AOK	irmovl	Y	0000000B	00000000	%eax	----

Execute Stage

E	Stat	Instr	valC	valA	valB	dstE	dstM	srcA	srcB
State	AOK	irmovl	0000000B	00000000	00000000	%eax	----	----	----
Input	AOK	irmovl	0000000C	00000000	00000000	%ebx	----	----	----

Decode Stage

D	Stat	Instr	rA	rB	valC	valP
State	AOK	irmovl	----	%ebx	0000000C	00000019
Input	AOK	addl	%eax	%ebx	00000000	0000001B

Register File

%eax	%ecx	%edx	%ebx	%esp	%ebp	%esi	%edi
1							

Stat AOK Condition Codes Z 1 S 0 O 0

Performance Cycles 2 Instructions 2 CPI 1.00

(a) Correct Implementation (std)

Program Code

File data_haz.yo Load

```
0x0  30f001000000              irmovl $1, %eax
0x6  30f302000000  W           irmovl $2, %ebx
0xc  10            M           nop
0xd  30f00b000000  E           irmovl $11, %eax
0x13 30f30c000000  D           irmovl $12, %ebx
0x19 6003          F           addl %eax, %ebx
0x1b 00                        halt #Should end with 23 in %ebx
```

Memory Contents

```
0x---0   0x---4   0x---8   0x---c
```

Y86 Processor: pipe-broken.hcl

Quit Go Stop Step Reset

Simulator Speed (10*log Hz) 0

Pipeline Registers

W	Stat	Instr	valE	valM	dstE	dstM
State	AOK	irmovl	00000002	00000000	%ebx	----
Input	AOK	nop	00000000	00000000	----	----

Memory Stage

M	Stat	Instr	Cnd	valE	valA	dstE	dstM
State	AOK	nop	Y	00000000	00000000	----	----
Input	AOK	irmovl	Y	0000000B	00000000	%eax	----

Execute Stage

E	Stat	Instr	valC	valA	valB	dstE	dstM	srcA	srcB
State	AOK	irmovl	0000000B	00000000	00000000	%eax	----	----	----
Input	AOK	irmovl	0000000C	00000000	00000000	%ebx	----	----	----

Decode Stage

D	Stat	Instr	rA	rB	valC	valP
State	AOK	irmovl	----	%ebx	0000000C	00000019
Input	AOK	addl	%eax	%ebx	00000000	0000001B

Register File

%eax	%ecx	%edx	%ebx	%esp	%ebp	%esi	%edi
1							

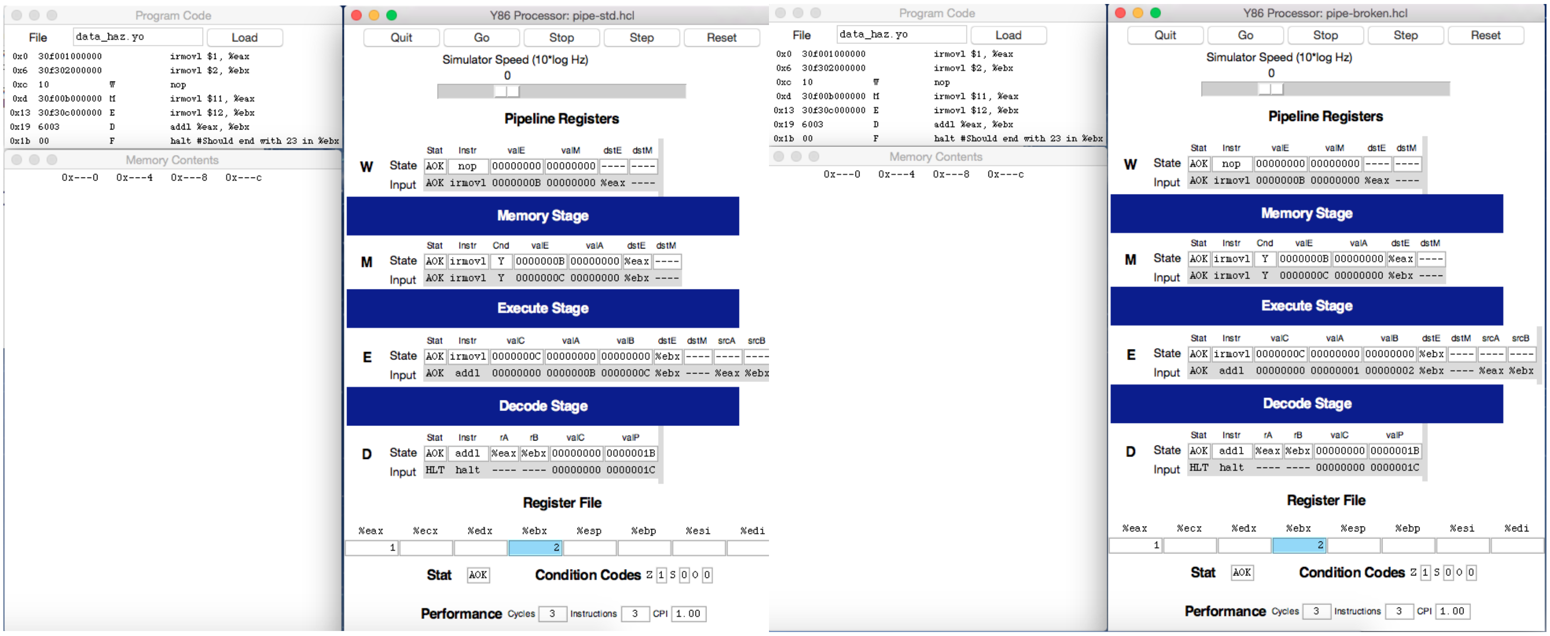
Stat AOK Condition Codes Z 1 S 0 O 0

Performance Cycles 2 Instructions 2 CPI 1.00

(b) Incorrect Implementation (broken)

Figure 1: In this step, the `addl %eax, %ebx` instruction has just been fetched by the two different versions. Note that there are two irmovl instructions writing to registers `%eax` and `%ebx` in the Execute and Decode Pipeline Registers respectively.

[*] COSC51 HW9 Q4



```
Program Code
File  data_haz.yo  Load
0x0   30f001000000       irmovl $1, %eax
0x6   30f302000000       irmovl $2, %ebx
0xc   10            W    nop
0xd   30f00b000000  M    irmovl $11, %eax
0x13  30f30c000000  E    irmovl $12, %ebx
0x19  6003          D    addl %eax, %ebx
0x1b  00            F    halt #Should end with 23 in %ebx

Memory Contents
        0x---0   0x---4   0x---8   0x---c
```

```
Y86 Processor: pipe-std.hcl
Quit   Go   Stop   Step   Reset
Simulator Speed (10*log Hz)
0

Pipeline Registers
          Stat  Instr   valE      valM      dstE  dstM
W  State  AOK   nop     00000000  00000000  ----  ----
   Input  AOK   irmovl  0000000B  00000000  %eax  ----
Memory Stage
          Stat  Instr   Cnd  valE      valA      dstE  dstM
M  State  AOK   irmovl  Y    0000000B  00000000  %eax  ----
   Input  AOK   irmovl  Y    0000000C  00000000  %ebx  ----
Execute Stage
          Stat  Instr   valC      valA      valB      dstE  dstM  srcA  srcB
E  State  AOK   irmovl  0000000C  00000000  00000000  %ebx  ----  ----  ----
   Input  AOK   addl    00000000  0000000B  0000000C  %ebx  ----  %eax  %ebx
Decode Stage
          Stat  Instr   rA    rB    valC      valP
D  State  AOK   addl    %eax  %ebx  00000000  0000001B
   Input  HLT   halt    ----  ----  00000000  0000001C

Register File
%eax   %ecx   %edx   %ebx   %esp   %ebp   %esi   %edi
   1                    2

Stat  AOK      Condition Codes  Z 1  S 0  O 0

Performance  Cycles 3  Instructions 3  CPI 1.00
```

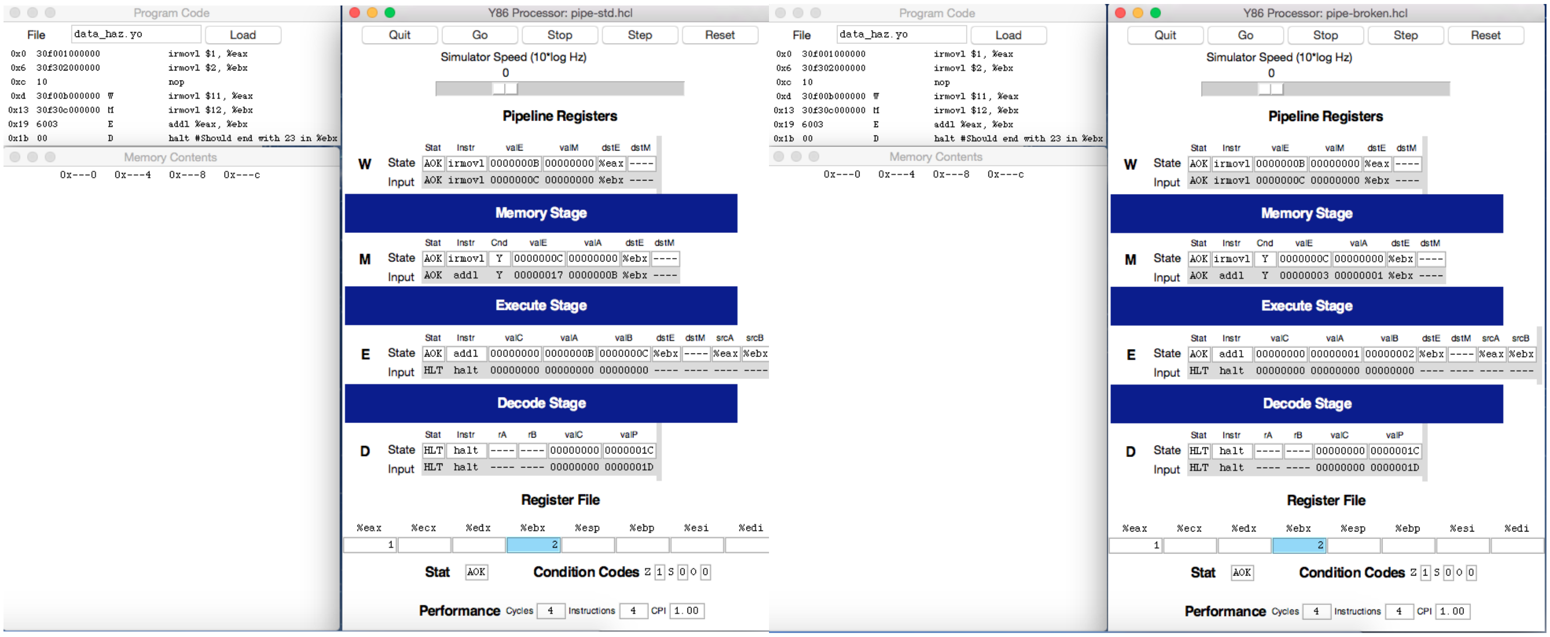
(a) Correct Implementation (std)

```
Program Code
File  data_haz.yo  Load
0x0   30f001000000       irmovl $1, %eax
0x6   30f302000000       irmovl $2, %ebx
0xc   10            W    nop
0xd   30f00b000000  M    irmovl $11, %eax
0x13  30f30c000000  E    irmovl $12, %ebx
0x19  6003          D    addl %eax, %ebx
0x1b  00            F    halt #Should end with 23 in %ebx

Memory Contents
        0x---0   0x---4   0x---8   0x---c
```

```
Y86 Processor: pipe-broken.hcl
Quit   Go   Stop   Step   Reset
Simulator Speed (10*log Hz)
0

Pipeline Registers
          Stat  Instr   valE      valM      dstE  dstM
W  State  AOK   nop     00000000  00000000  ----  ----
   Input  AOK   irmovl  0000000B  00000000  %eax  ----
Memory Stage
          Stat  Instr   Cnd  valE      valA      dstE  dstM
M  State  AOK   irmovl  Y    0000000B  00000000  %eax  ----
   Input  AOK   irmovl  Y    0000000C  00000000  %ebx  ----
Execute Stage
          Stat  Instr   valC      valA      valB      dstE  dstM  srcA  srcB
E  State  AOK   irmovl  0000000C  00000000  00000000  %ebx  ----  ----  ----
   Input  AOK   addl    00000000  00000001  00000002  %ebx  ----  %eax  %ebx
Decode Stage
          Stat  Instr   rA    rB    valC      valP
D  State  AOK   addl    %eax  %ebx  00000000  0000001B
   Input  HLT   halt    ----  ----  00000000  0000001C

Register File
%eax   %ecx   %edx   %ebx   %esp   %ebp   %esi   %edi
   1                    2

Stat  AOK      Condition Codes  Z 1  S 0  O 0

Performance  Cycles 3  Instructions 3  CPI 1.00
```

(b) Incorrect Implemenetation (broken)

Figure 2: In this step, both implementations' programmer-visible states are still identical as the *addl* instruction is moved into the Decode Pipeline Register.



```
Program Code
File  data_haz.yo  Load
0x0   30f001000000       irmovl $1, %eax
0x6   30f302000000       irmovl $2, %ebx
0xc   10                 nop
0xd   30f00b000000  W    irmovl $11, %eax
0x13  30f30c000000  M    irmovl $12, %ebx
0x19  6003          E    addl %eax, %ebx
0x1b  00            D    halt #Should end with 23 in %ebx

Memory Contents
        0x---0   0x---4   0x---8   0x---c
```

```
Y86 Processor: pipe-std.hcl
Quit   Go   Stop   Step   Reset
Simulator Speed (10*log Hz)
0

Pipeline Registers
          Stat  Instr   valE      valM      dstE  dstM
W  State  AOK   irmovl  0000000B  00000000  %eax  ----
   Input  AOK   irmovl  0000000C  00000000  %ebx  ----
Memory Stage
          Stat  Instr   Cnd  valE      valA      dstE  dstM
M  State  AOK   irmovl  Y    0000000C  00000000  %ebx  ----
   Input  AOK   addl    Y    00000017  0000000B  %ebx  ----
Execute Stage
          Stat  Instr   valC      valA      valB      dstE  dstM  srcA  srcB
E  State  AOK   addl    00000000  0000000B  0000000C  %ebx  ----  %eax  %ebx
   Input  HLT   halt    00000000  00000000  00000000  ----  ----  ----  ----
Decode Stage
          Stat  Instr   rA    rB    valC      valP
D  State  HLT   halt    ----  ----  00000000  0000001C
   Input  HLT   halt    ----  ----  00000000  0000001D

Register File
%eax   %ecx   %edx   %ebx   %esp   %ebp   %esi   %edi
   1                    2

Stat  AOK      Condition Codes  Z 1  S 0  O 0

Performance  Cycles 4  Instructions 4  CPI 1.00
```

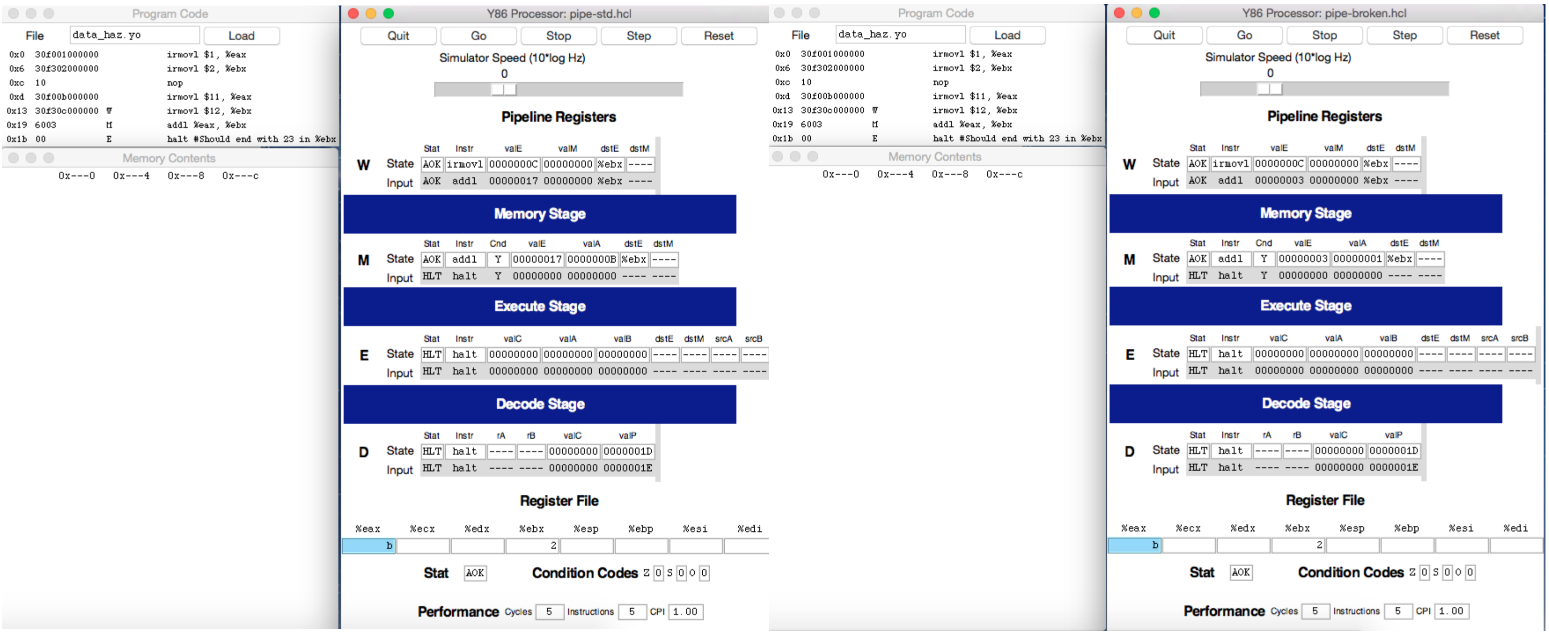
(a) Correct Implementation (std)

```
Program Code
File  data_haz.yo  Load
0x0   30f001000000       irmovl $1, %eax
0x6   30f302000000       irmovl $2, %ebx
0xc   10                 nop
0xd   30f00b000000  W    irmovl $11, %eax
0x13  30f30c000000  M    irmovl $12, %ebx
0x19  6003          E    addl %eax, %ebx
0x1b  00            D    halt #Should end with 23 in %ebx

Memory Contents
        0x---0   0x---4   0x---8   0x---c
```

```
Y86 Processor: pipe-broken.hcl
Quit   Go   Stop   Step   Reset
Simulator Speed (10*log Hz)
0

Pipeline Registers
          Stat  Instr   valE      valM      dstE  dstM
W  State  AOK   irmovl  0000000B  00000000  %eax  ----
   Input  AOK   irmovl  0000000C  00000000  %ebx  ----
Memory Stage
          Stat  Instr   Cnd  valE      valA      dstE  dstM
M  State  AOK   irmovl  Y    0000000C  00000000  %ebx  ----
   Input  AOK   addl    Y    00000003  00000001  %ebx  ----
Execute Stage
          Stat  Instr   valC      valA      valB      dstE  dstM  srcA  srcB
E  State  AOK   addl    00000000  00000001  00000002  %ebx  ----  %eax  %ebx
   Input  HLT   halt    00000000  00000000  00000000  ----  ----  ----  ----
Decode Stage
          Stat  Instr   rA    rB    valC      valP
D  State  HLT   halt    ----  ----  00000000  0000001C
   Input  HLT   halt    ----  ----  00000000  0000001D

Register File
%eax   %ecx   %edx   %ebx   %esp   %ebp   %esi   %edi
   1                    2

Stat  AOK      Condition Codes  Z 1  S 0  O 0

Performance  Cycles 4  Instructions 4  CPI 1.00
```

(b) Incorrect Implementation (broken)

Figure 3: Here we can clearly see where the data hazard arises. When the correct implementation decoded `rA` and `rB`, it forwarded the values still in the process of being written by the two `irmovl` instructions still in the Write and Memory Pipeline Registers.
In contrast, the incorrect implementation decoded `rA` and `rB` to be the values written by the `irmovl` instructions from lines 1 and 2 (which have already been written). As a result, there's a disparity between the `valA` and `valB` values in the Execute Pipeline Registers of the two implementations. The incorrect version is using outdated values as its operands.



```
Program Code
File data_haz.yo                    Load
0x0   30f001000000       irmovl $1, %eax
0x6   30f302000000       irmovl $2, %ebx
0xc   10                 nop
0xd   30f00b000000       irmovl $11, %eax
0x13  30f30c000000  W    irmovl $12, %ebx
0x19  6003          M    addl %eax, %ebx
0x1b  00            E    halt #Should end with 23 in %ebx

Memory Contents
      0x---0   0x---4   0x---8   0x---c
```

```
Y86 Processor: pipe-std.hcl
Quit   Go   Stop   Step   Reset
Simulator Speed (10*log Hz)
0

Pipeline Registers
             Stat  Instr   valE      valM      dstE  dstM
W   State    AOK   irmovl  0000000C  00000000  %ebx  ----
    Input    AOK   addl    00000017  00000000  %ebx  ----

Memory Stage
             Stat  Instr   Cnd  valE      valA      dstE  dstM
M   State    AOK   addl    Y    00000017  0000000B  %ebx  ----
    Input    HLT   halt    Y    00000000  00000000  ----  ----

Execute Stage
             Stat  Instr   valC      valA      valB      dstE  dstM  srcA  srcB
E   State    HLT   halt    00000000  00000000  00000000  ----  ----  ----  ----
    Input    HLT   halt    00000000  00000000  00000000  ----  ----  ----  ----

Decode Stage
             Stat  Instr   rA    rB    valC      valP
D   State    HLT   halt    ----  ----  00000000  0000001D
    Input    HLT   halt    ----  ----  00000000  0000001E

Register File
%eax   %ecx   %edx   %ebx   %esp   %ebp   %esi   %edi
   b                    2

Stat AOK     Condition Codes Z 0 S 0 O 0

Performance Cycles 5 Instructions 5 CPI 1.00
```

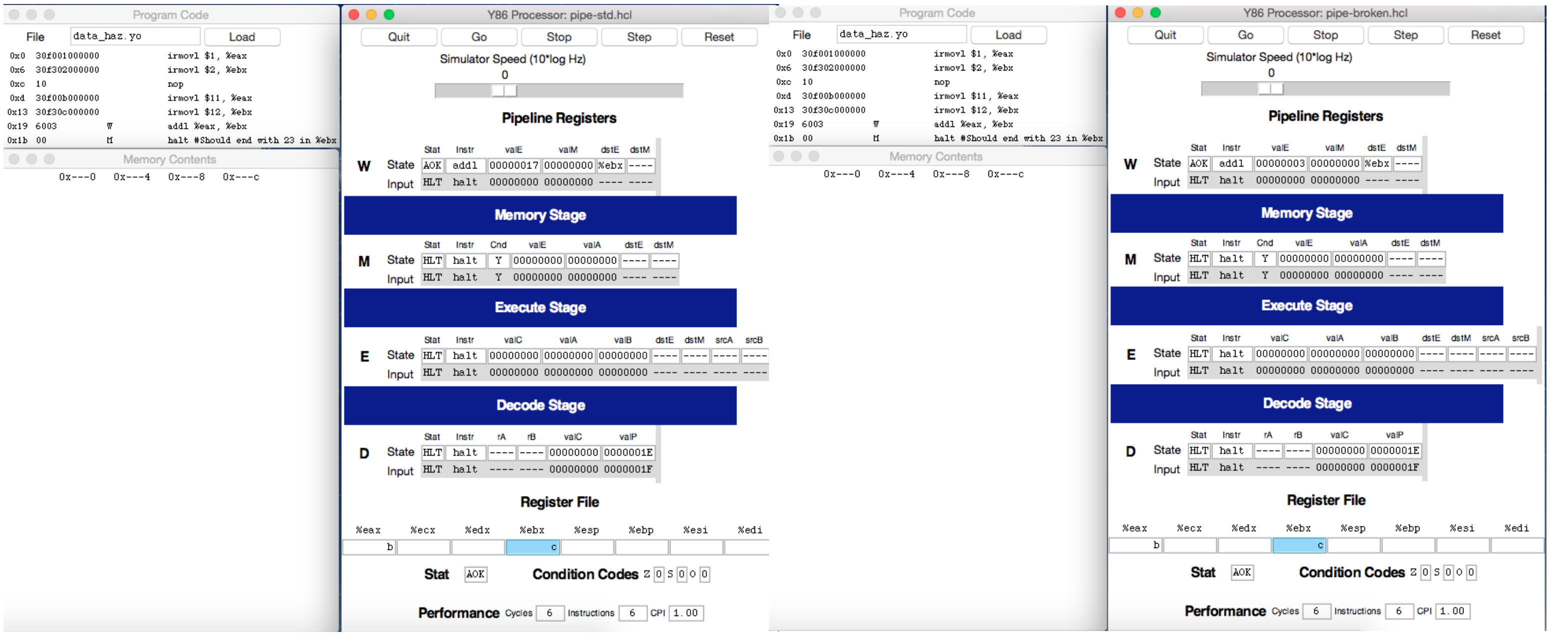
(a) Correct Implementation (std)

```
Program Code
File data_haz.yo                    Load
0x0   30f001000000       irmovl $1, %eax
0x6   30f302000000       irmovl $2, %ebx
0xc   10                 nop
0xd   30f00b000000       irmovl $11, %eax
0x13  30f30c000000  W    irmovl $12, %ebx
0x19  6003          M    addl %eax, %ebx
0x1b  00            E    halt #Should end with 23 in %ebx

Memory Contents
      0x---0   0x---4   0x---8   0x---c
```

```
Y86 Processor: pipe-broken.hcl
Quit   Go   Stop   Step   Reset
Simulator Speed (10*log Hz)
0

Pipeline Registers
             Stat  Instr   valE      valM      dstE  dstM
W   State    AOK   irmovl  0000000C  00000000  %ebx  ----
    Input    AOK   addl    00000003  00000000  %ebx  ----

Memory Stage
             Stat  Instr   Cnd  valE      valA      dstE  dstM
M   State    AOK   addl    Y    00000003  00000001  %ebx  ----
    Input    HLT   halt    Y    00000000  00000000  ----  ----

Execute Stage
             Stat  Instr   valC      valA      valB      dstE  dstM  srcA  srcB
E   State    HLT   halt    00000000  00000000  00000000  ----  ----  ----  ----
    Input    HLT   halt    00000000  00000000  00000000  ----  ----  ----  ----

Decode Stage
             Stat  Instr   rA    rB    valC      valP
D   State    HLT   halt    ----  ----  00000000  0000001D
    Input    HLT   halt    ----  ----  00000000  0000001E

Register File
%eax   %ecx   %edx   %ebx   %esp   %ebp   %esi   %edi
   b                    2

Stat AOK     Condition Codes Z 0 S 0 O 0

Performance Cycles 5 Instructions 5 CPI 1.00
```

(b) Incorrect Implementation (broken)

Figure 4: In this state, the `irmovl` instruction writing to register `%eax` goes the the write-back stage. The `addl` instruction moves into the Memory Pipeline Register. Notice the correct value of the sum is in Figure (a) `valE`, while an incorrect value of the sum is in Figure (b) `valE`.



```
Program Code
File data_haz.yo                    Load
0x0   30f001000000       irmovl $1, %eax
0x6   30f302000000       irmovl $2, %ebx
0xc   10                 nop
0xd   30f00b000000       irmovl $11, %eax
0x13  30f30c000000       irmovl $12, %ebx
0x19  6003          W    addl %eax, %ebx
0x1b  00            M    halt #Should end with 23 in %ebx

Memory Contents
      0x---0   0x---4   0x---8   0x---c
```

```
Y86 Processor: pipe-std.hcl
Quit   Go   Stop   Step   Reset
Simulator Speed (10*log Hz)
0

Pipeline Registers
             Stat  Instr   valE      valM      dstE  dstM
W   State    AOK   addl    00000017  00000000  %ebx  ----
    Input    HLT   halt    00000000  00000000  ----  ----

Memory Stage
             Stat  Instr   Cnd  valE      valA      dstE  dstM
M   State    HLT   halt    Y    00000000  00000000  ----  ----
    Input    HLT   halt    Y    00000000  00000000  ----  ----

Execute Stage
             Stat  Instr   valC      valA      valB      dstE  dstM  srcA  srcB
E   State    HLT   halt    00000000  00000000  00000000  ----  ----  ----  ----
    Input    HLT   halt    00000000  00000000  00000000  ----  ----  ----  ----

Decode Stage
             Stat  Instr   rA    rB    valC      valP
D   State    HLT   halt    ----  ----  00000000  0000001E
    Input    HLT   halt    ----  ----  00000000  0000001F

Register File
%eax   %ecx   %edx   %ebx   %esp   %ebp   %esi   %edi
   b                    c

Stat AOK     Condition Codes Z 0 S 0 O 0

Performance Cycles 6 Instructions 6 CPI 1.00
```

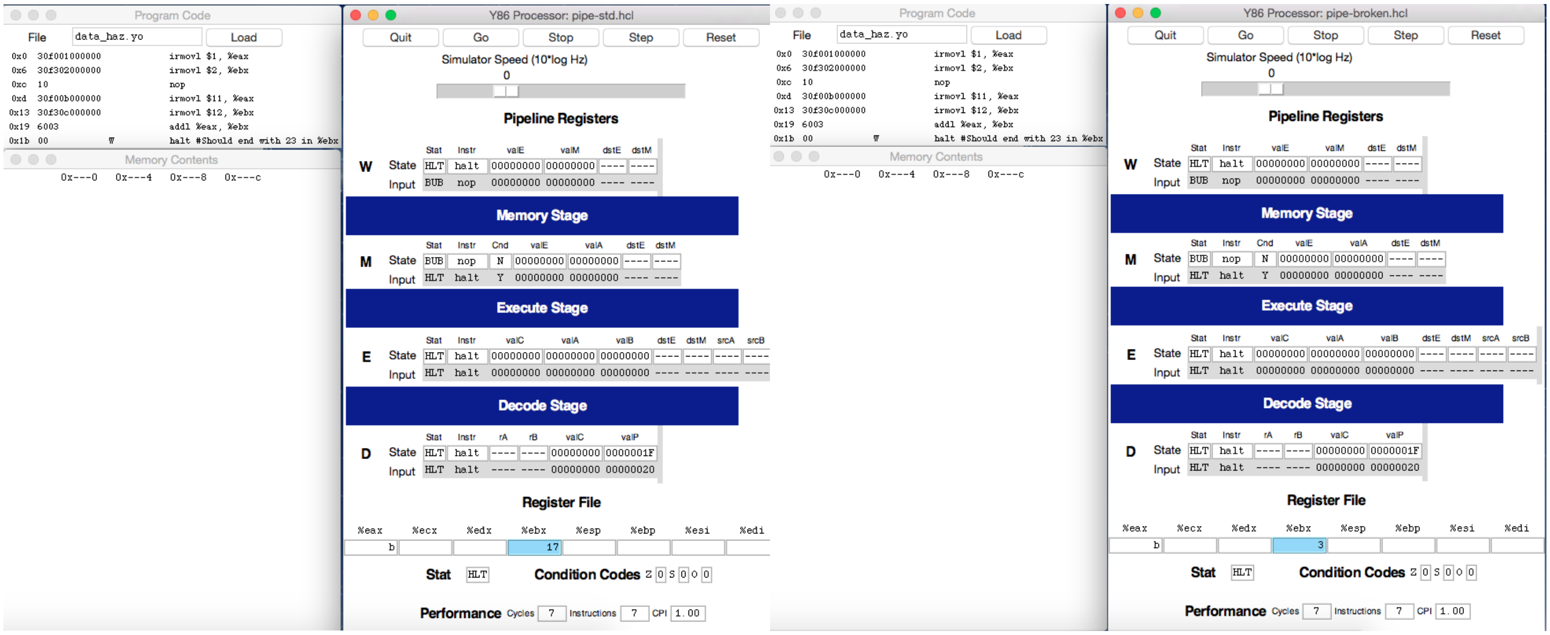
(a) Correct Implementation (std)

```
Program Code
File data_haz.yo                    Load
0x0   30f001000000       irmovl $1, %eax
0x6   30f302000000       irmovl $2, %ebx
0xc   10                 nop
0xd   30f00b000000       irmovl $11, %eax
0x13  30f30c000000       irmovl $12, %ebx
0x19  6003          W    addl %eax, %ebx
0x1b  00            M    halt #Should end with 23 in %ebx

Memory Contents
      0x---0   0x---4   0x---8   0x---c
```

```
Y86 Processor: pipe-broken.hcl
Quit   Go   Stop   Step   Reset
Simulator Speed (10*log Hz)
0

Pipeline Registers
             Stat  Instr   valE      valM      dstE  dstM
W   State    AOK   addl    00000003  00000000  %ebx  ----
    Input    HLT   halt    00000000  00000000  ----  ----

Memory Stage
             Stat  Instr   Cnd  valE      valA      dstE  dstM
M   State    HLT   halt    Y    00000000  00000000  ----  ----
    Input    HLT   halt    Y    00000000  00000000  ----  ----

Execute Stage
             Stat  Instr   valC      valA      valB      dstE  dstM  srcA  srcB
E   State    HLT   halt    00000000  00000000  00000000  ----  ----  ----  ----
    Input    HLT   halt    00000000  00000000  00000000  ----  ----  ----  ----

Decode Stage
             Stat  Instr   rA    rB    valC      valP
D   State    HLT   halt    ----  ----  00000000  0000001E
    Input    HLT   halt    ----  ----  00000000  0000001F

Register File
%eax   %ecx   %edx   %ebx   %esp   %ebp   %esi   %edi
   b                    c

Stat AOK     Condition Codes Z 0 S 0 O 0

Performance Cycles 6 Instructions 6 CPI 1.00
```

(b) Incorrect Implementation (broken)

Figure 5: The `addl` instruction moves into the Write Pipeline Register.



File: data_haz.yo

```
0x0  30f001000000        irmovl $1, %eax
0x6  30f302000000        irmovl $2, %ebx
0xc  10                  nop
0xd  30f00b000000        irmovl $11, %eax
0x13 30f30c000000        irmovl $12, %ebx
0x19 6003                addl %eax, %ebx
0x1b 00            W     halt #Should end with 23 in %ebx
```

```
0x---0  0x---4  0x---8  0x---c
```

Simulator Speed (10*log Hz)
0

Pipeline Registers

W	Stat	Instr	valE	valM	dstE	dstM
State	HLT	halt	00000000	00000000	----	----
Input	BUB	nop	00000000	00000000	----	----

M	Stat	Instr	Cnd	valE	valA	dstE	dstM
State	BUB	nop	N	00000000	00000000	----	----
Input	HLT	halt	Y	00000000	00000000	----	----

E	Stat	Instr	valC	valA	valB	dstE	dstM	srcA	srcB
State	HLT	halt	00000000	00000000	00000000	----	----	----	----
Input	HLT	halt	00000000	00000000	00000000	----	----	----	----

D	Stat	Instr	rA	rB	valC	valP
State	HLT	halt	----	----	00000000	0000001F
Input	HLT	halt	----	----	00000000	00000020

Register File

%eax	%ecx	%edx	%ebx	%esp	%ebp	%esi	%edi
b			17				

Stat HLT Condition Codes Z 0 S 0 O 0

Performance Cycles 7 Instructions 7 CPI 1.00

(a) Correct Implementation (std)

File: data_haz.yo

```
0x0  30f001000000        irmovl $1, %eax
0x6  30f302000000        irmovl $2, %ebx
0xc  10                  nop
0xd  30f00b000000        irmovl $11, %eax
0x13 30f30c000000        irmovl $12, %ebx
0x19 6003                addl %eax, %ebx
0x1b 00            W     halt #Should end with 23 in %ebx
```

```
0x---0  0x---4  0x---8  0x---c
```

Simulator Speed (10*log Hz)
0

Pipeline Registers

W	Stat	Instr	valE	valM	dstE	dstM
State	HLT	halt	00000000	00000000	----	----
Input	BUB	nop	00000000	00000000	----	----

M	Stat	Instr	Cnd	valE	valA	dstE	dstM
State	BUB	nop	N	00000000	00000000	----	----
Input	HLT	halt	Y	00000000	00000000	----	----

E	Stat	Instr	valC	valA	valB	dstE	dstM	srcA	srcB
State	HLT	halt	00000000	00000000	00000000	----	----	----	----
Input	HLT	halt	00000000	00000000	00000000	----	----	----	----

D	Stat	Instr	rA	rB	valC	valP
State	HLT	halt	----	----	00000000	0000001F
Input	HLT	halt	----	----	00000000	00000020

Register File

%eax	%ecx	%edx	%ebx	%esp	%ebp	%esi	%edi
b			3				

Stat HLT Condition Codes Z 0 S 0 O 0

Performance Cycles 7 Instructions 7 CPI 1.00

(b) Incorrect Implementation (broken)

Figure 6: The sum calculated in the `addl` instruction is written back to register `%eax`. As you can see, due to the data hazard, the correct sum of $11 + 12 = 23$ (in hex, `0x17`) is calculated by the correct pipelining implementation and written to register `%eax` in Figure (a).
On the other hand, you can see the incorrect sum of $1 + 2 = 3$ is calculated by the incorrect pipelining implementation and is written to register `%eax` in Figure (b). This displays a data hazard.