ELE 455/555
Computer System Engineering

Section 1 – Review and Foundations

Class 4 – Architecture and Operation

Architecture

Classes of Processors

- General Purpose Processor
  - User Programmable
    - Intended to run end user selected programs
  - Application Independent
    - PowerPoint, Chrome, Twitter, Angry birds, …
- Embedded Processor
  - Not User Programmable
    - Programmed by manufacturer
  - Application Driven
    - Non-smart phone, appliances, missiles, automobiles, …
    - Very wide and very deep applications profile

Architecture

Classes of Processors

- General Purpose Processor
  - Key Characteristics
    - 32/64 bit operations
    - Support non-real-time/time-sharing operating systems
    - Support complex memory systems
      - Multi-level cache
      - dRAM
      - Virtual memory
    - Support DMA-driven I/O
    - Complex CPU structures
      - Pipelining
      - Superscalar execution
      - Out-of-order execution (OOO)
      - Floating Point HW

Architecture

Classes of Processors

- General Purpose Processor
  - Examples
    - ARM 7, 9, Cortex A8, A9,A15
    - Intel Pentiums, Ix, …
    - AMD Phenom, Athleron, Opteron
    - Apple A4, A5
    - TI OMAPs

Architecture

Classes of Processors

- Embedded Processor
  - Key Characteristics
    - 4/8/16/32 bit operations
    - Support real-time operating systems
    - Relatively simple memory systems
    - Memory mapped I/O
    - Simple CPU structures
      - Few registers
      - Limited Instructions
    - Support for multiple I/O schemes
    - Wide range of peripheral support
      - A/D – D/A
      - Sensors
      - Extensive interrupt support

Architecture

Classes of Processors

- Embedded Processor
  - Examples
    - Motorola/Freescale 68K, HC11, HCS12
    - ARM Cortex Rx, Mx
    - Atmel AVR

Architecture

Instruction Sets

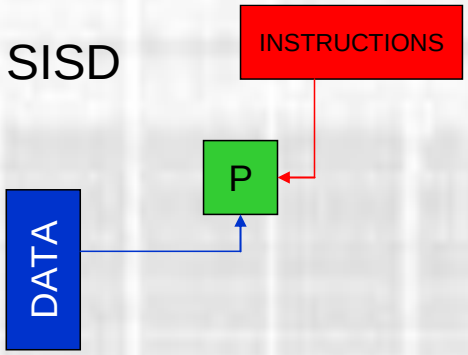
- CISC – Complex Instruction Set Computer
  - Name didn't even exist until RISC was defined
  - Used in most processors until about 1980
  - One instruction holds multiple actions
    - Load data from location, add, write data to new location
  - Many times the instructions were designed to emulate high level language constructs
- RISC – Reduced Instruction Set Computer
  - Developed in the '80s
  - Most prevalent architecture today
  - Sometimes called a load/store architecture
  - Instructions are simple
    - Load data from location
    - Add
    - Store data to location
- RISC dominates today
  - Much easier to take advantage of advanced structures like Pipelining, Superscalar, OOO

Architecture

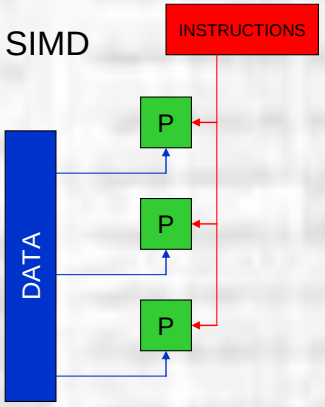
Architectural Configurations

- Instruction / Data Structures

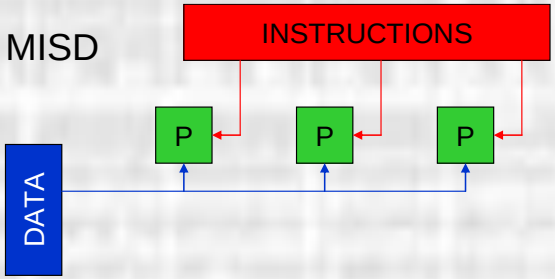
SISD – Single instruction – Single Data



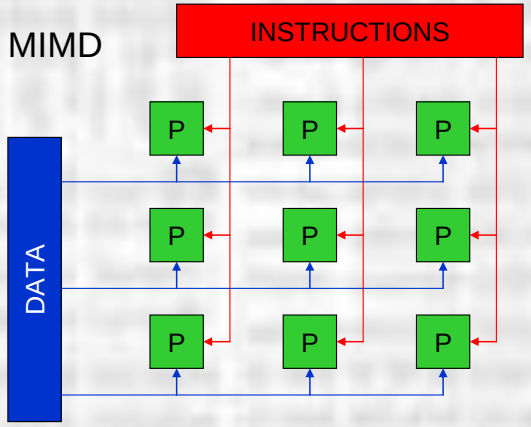
SIMD – Single Instruction – Multiple Data



MISD – Multiple Instruction – Single Data



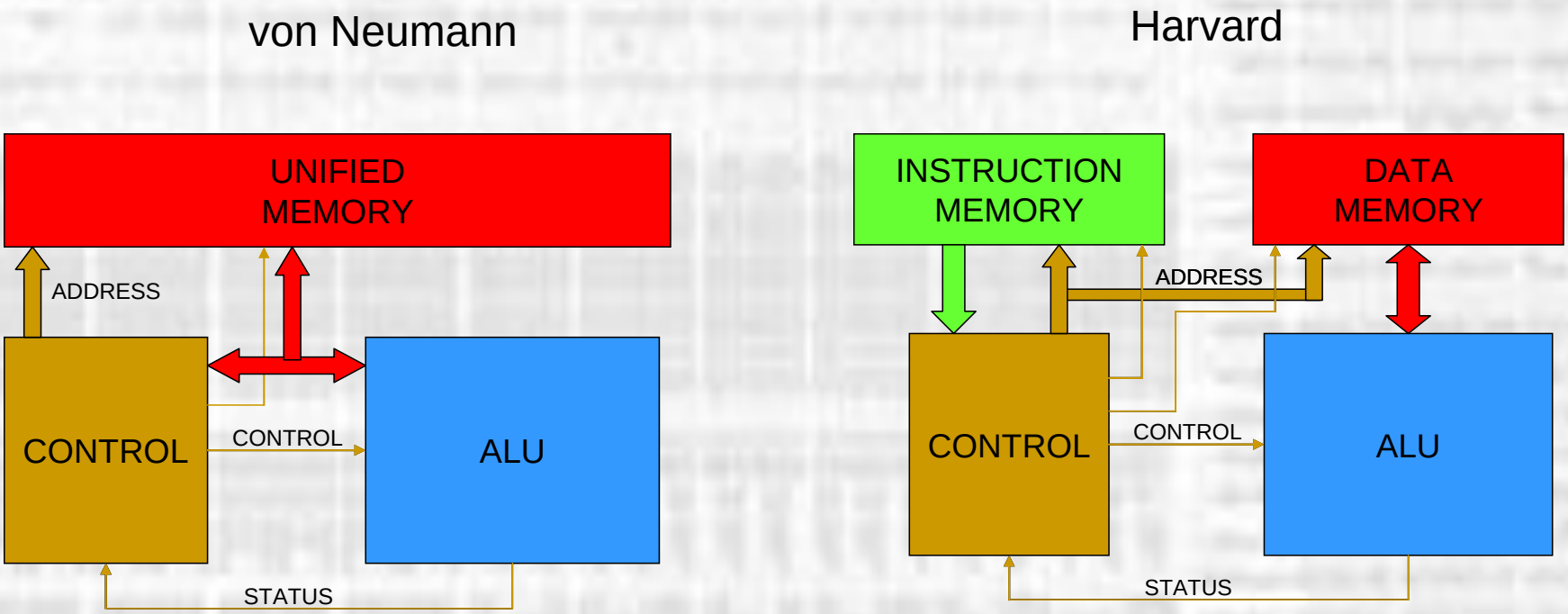
•MIMD – Multiple Instruction – Multiple Data



Architecture

Architectural Configurations

- Memory Bus Structure

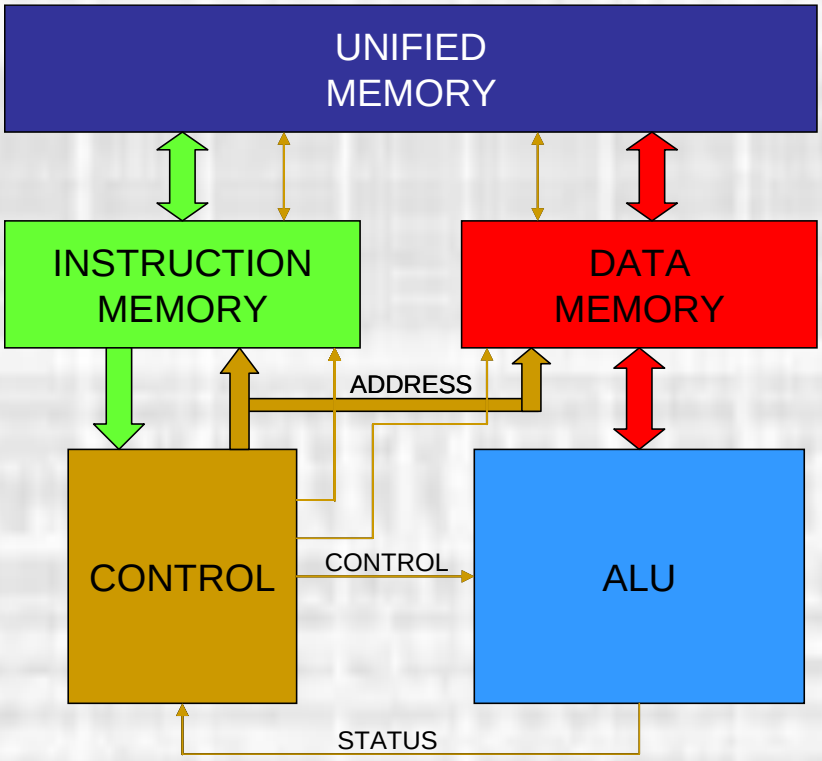


Architecture

Architectural Configurations

- Memory Bus Structure

Modified Harvard

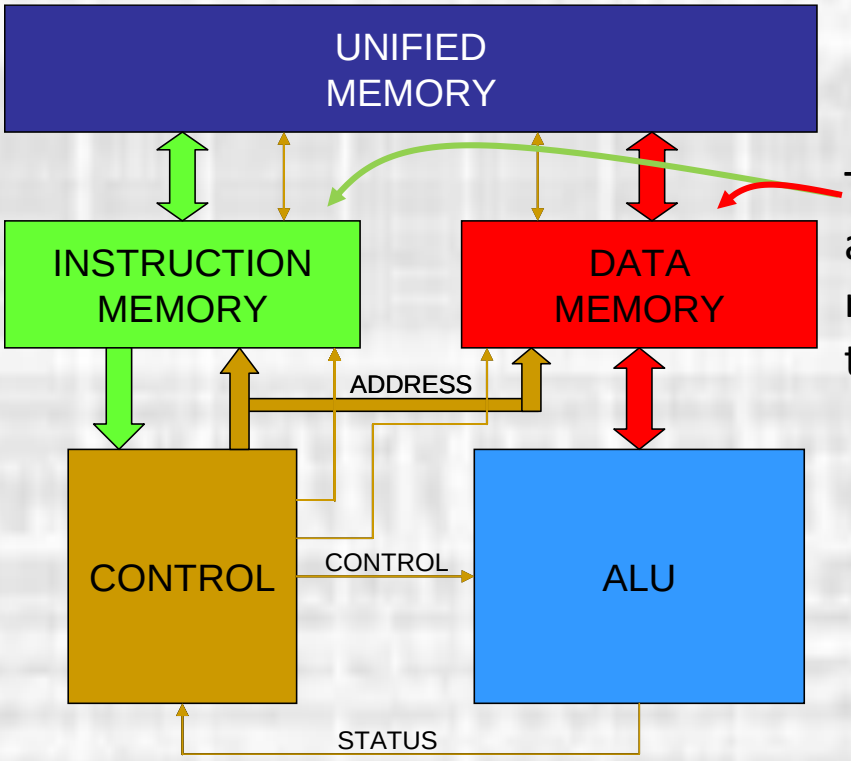


Architecture

Architectural Configurations

- Cache Memory

Modified Harvard



These memories are often augmented by cache memories or are caches themselves

Architecture

Architectural Configurations

- Cache Memory
  - Cache memory is used to store relatively small amounts of data or program for a relatively short amount of time
  - Sit between the processor and the main memory
  - Fast – keep them small to make them fast – allow the processor to run faster than main memory would allow
  - Leverage the concept of temporal locality
    - If you have recently used a piece of data you are more likely to use it again
  - Leverage the concept of spatial locality
    - Program code and data structures are generally contiguous in memory

Architecture

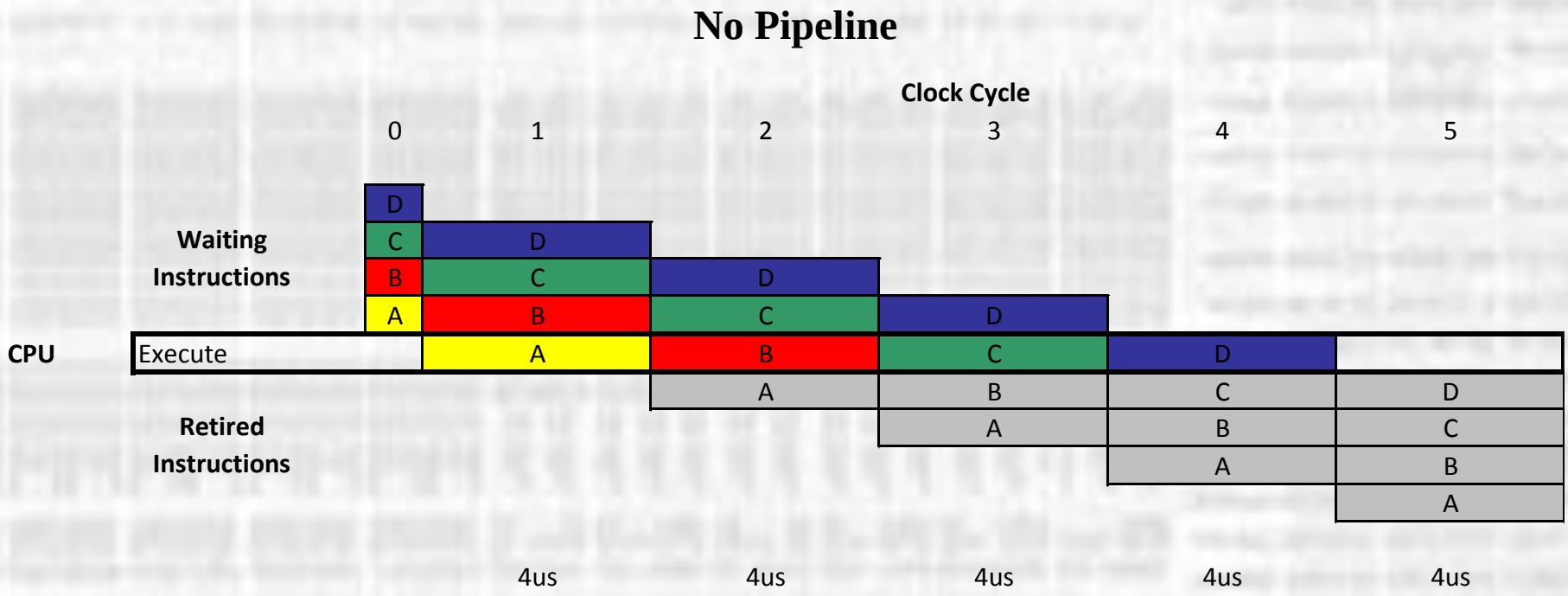
Architectural Configurations

- Cache Memory
  - Basic Operation
    - Processor requests a byte of program or data
    - The system first checks to see if the byte is already in the cache
    - if Yes – read the byte and continue (called a cache hit)
    - if No
      - stall or allow the processor to do something else (called a cache miss)
      - read the byte from main memory into the cache
      - read the byte from the cache and continue
    - If the cache is full and a new byte needs to be loaded – several methods can be used to remove an existing byte
      - LRU – least recently used byte is removed
      - FIF0 – oldest byte loaded is removed

Architecture

Architectural Configurations

- Pipelining

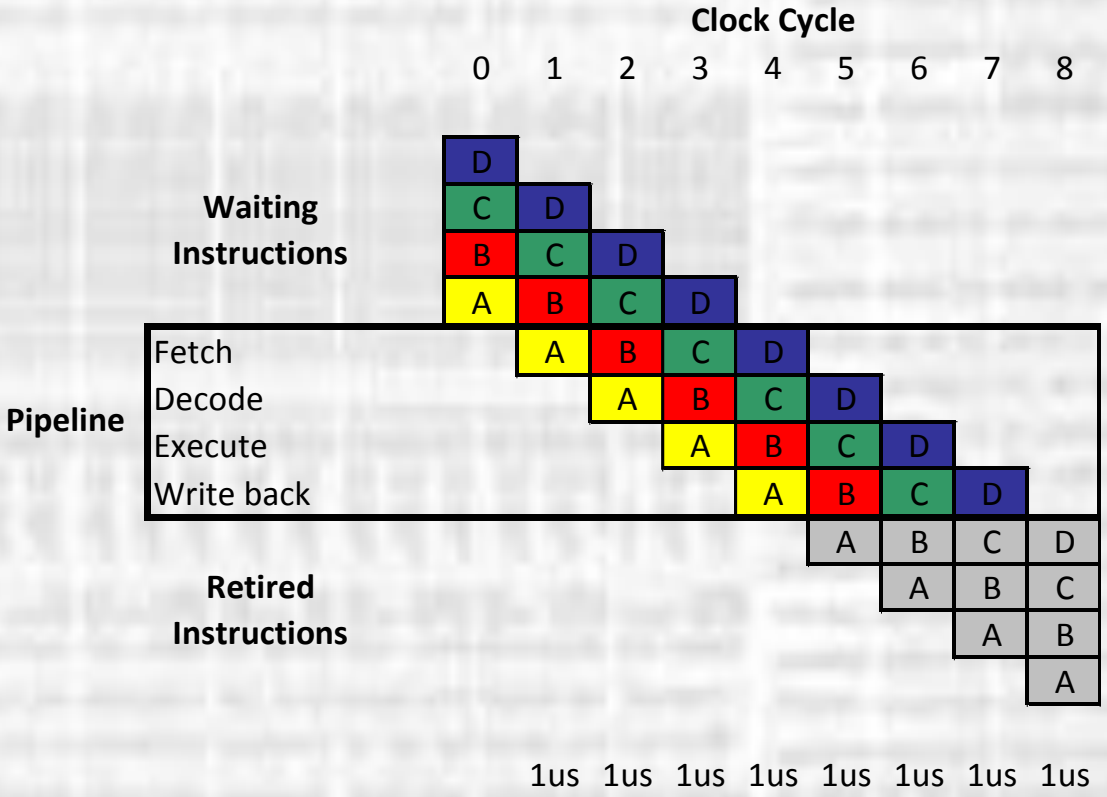


Execute = fetch instruction, decode, execute, write back

Architecture

Architectural Configurations

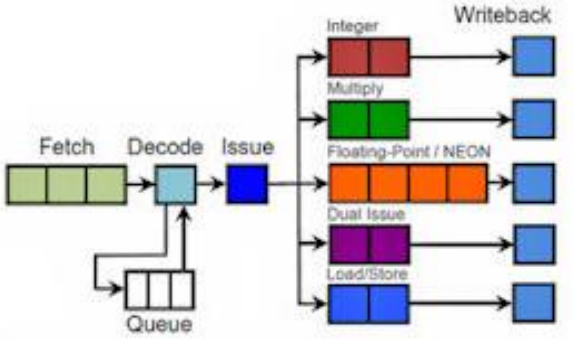
- Pipelining
  - Break complex tasks into smaller chunks
  - Start the next instruction as soon as each subtask is complete



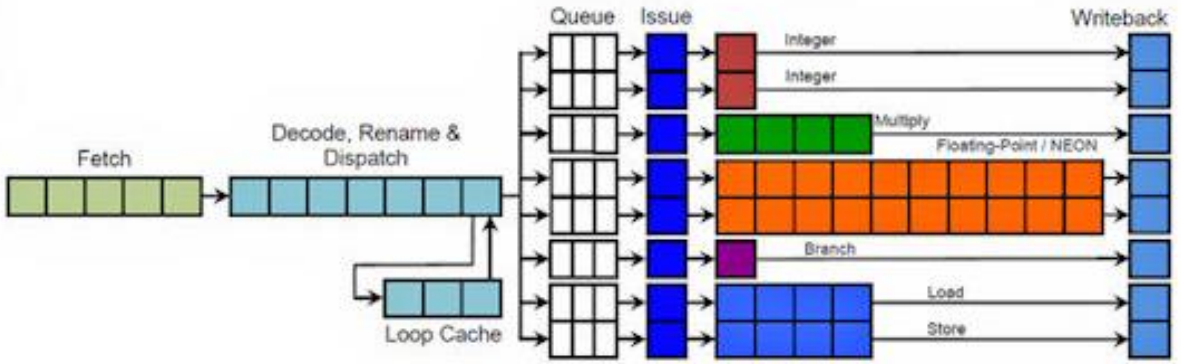
Architecture

Architectural Configurations

- Superscalar
  - Parallelism at the micro-architecture level



ARM Cortex-A7 Pipeline

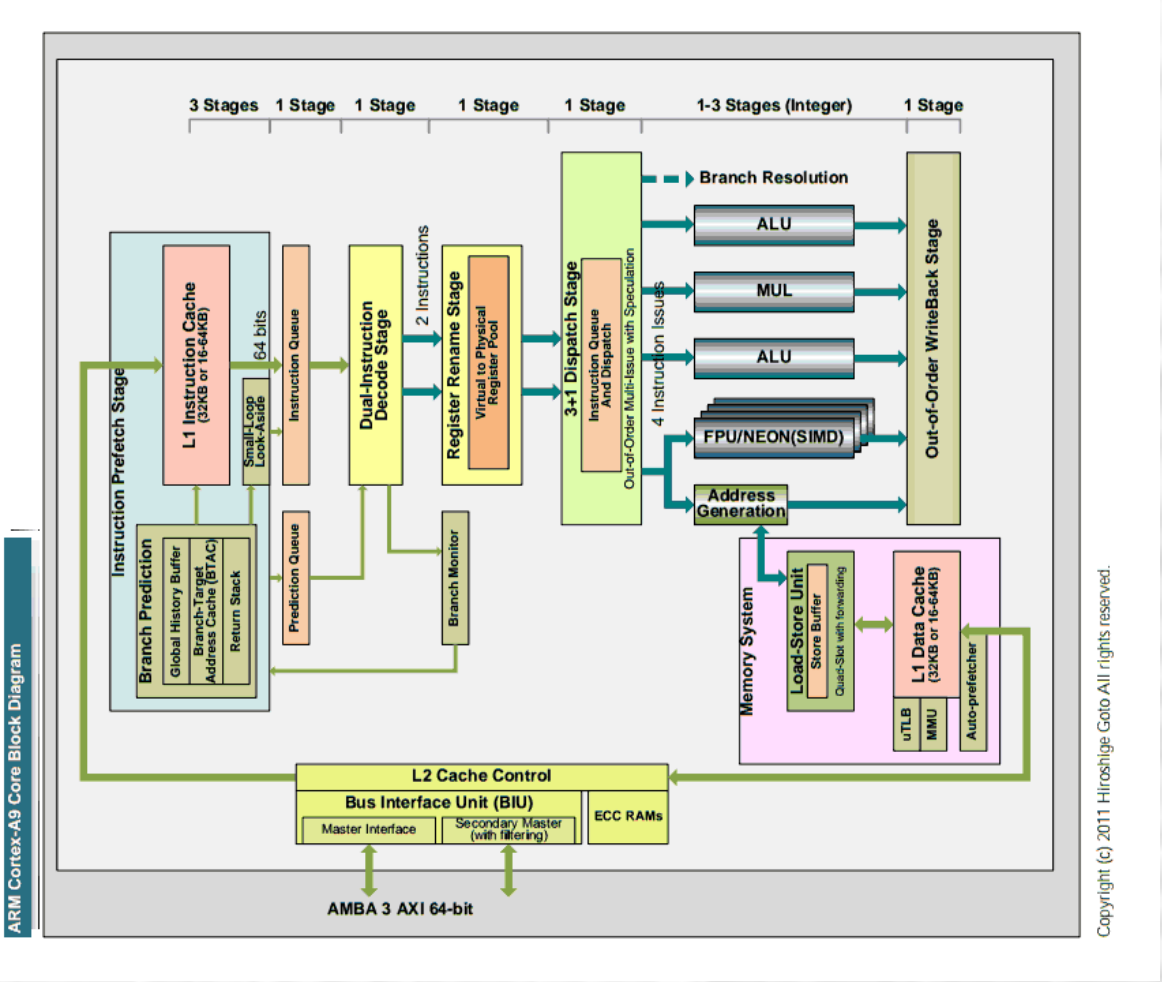


ARM Cortex-A15 Pipeline

Architecture

Architectural Configurations

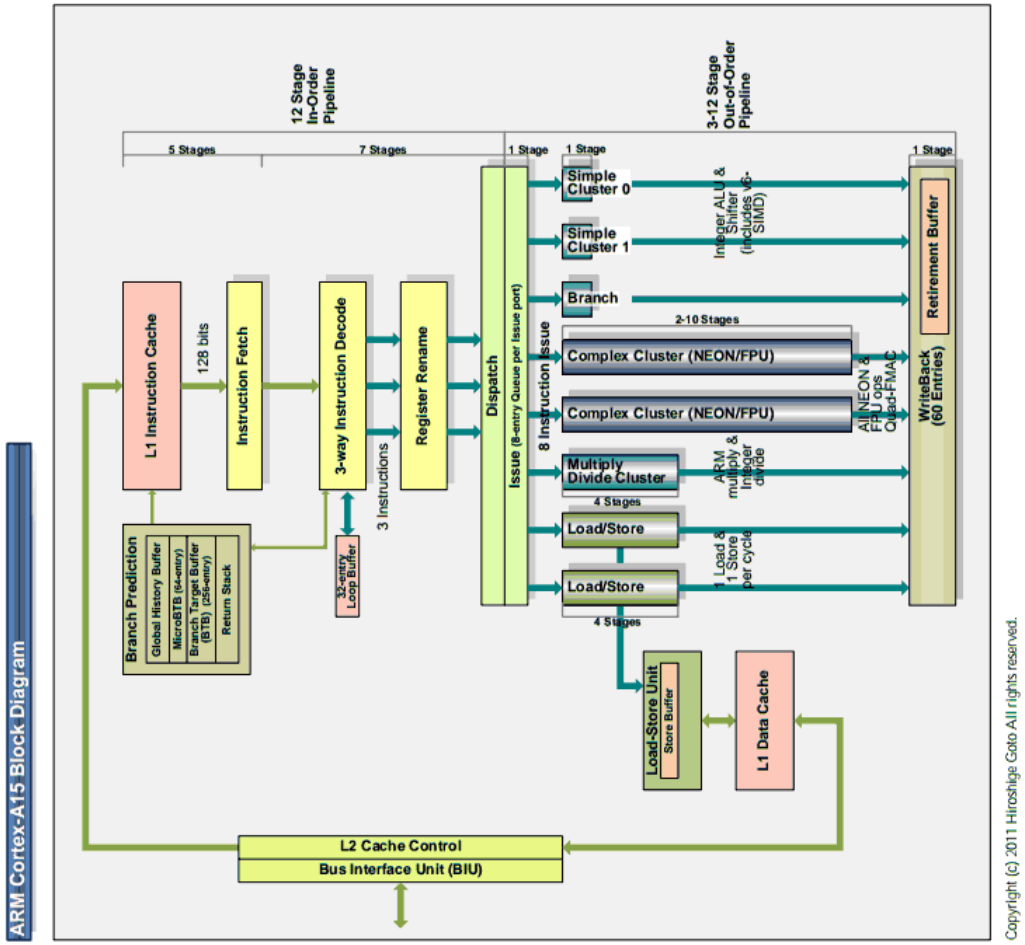
- Modern Example



Architecture

Architectural Configurations

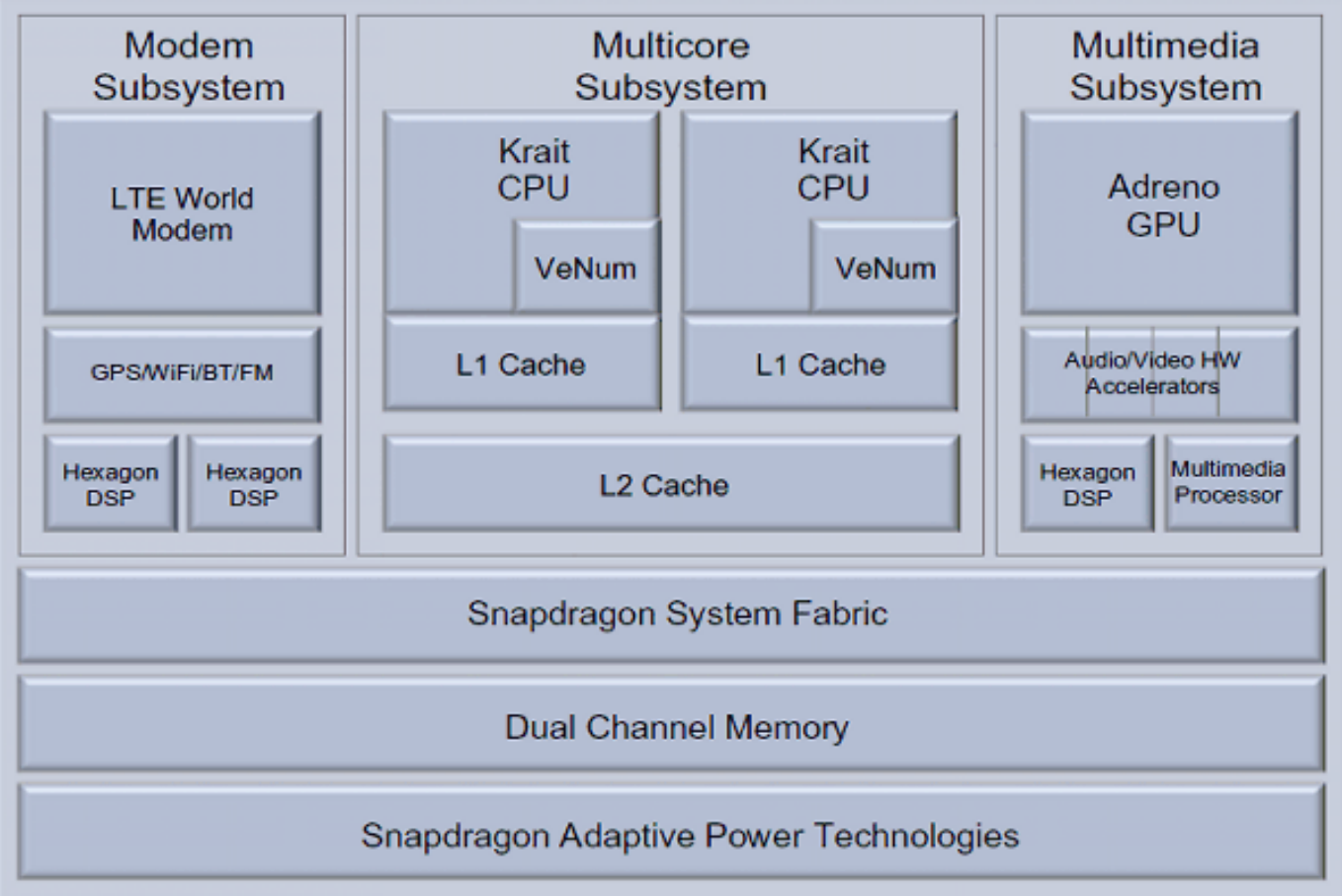
- Modern Example



Architecture

Architectural Configurations

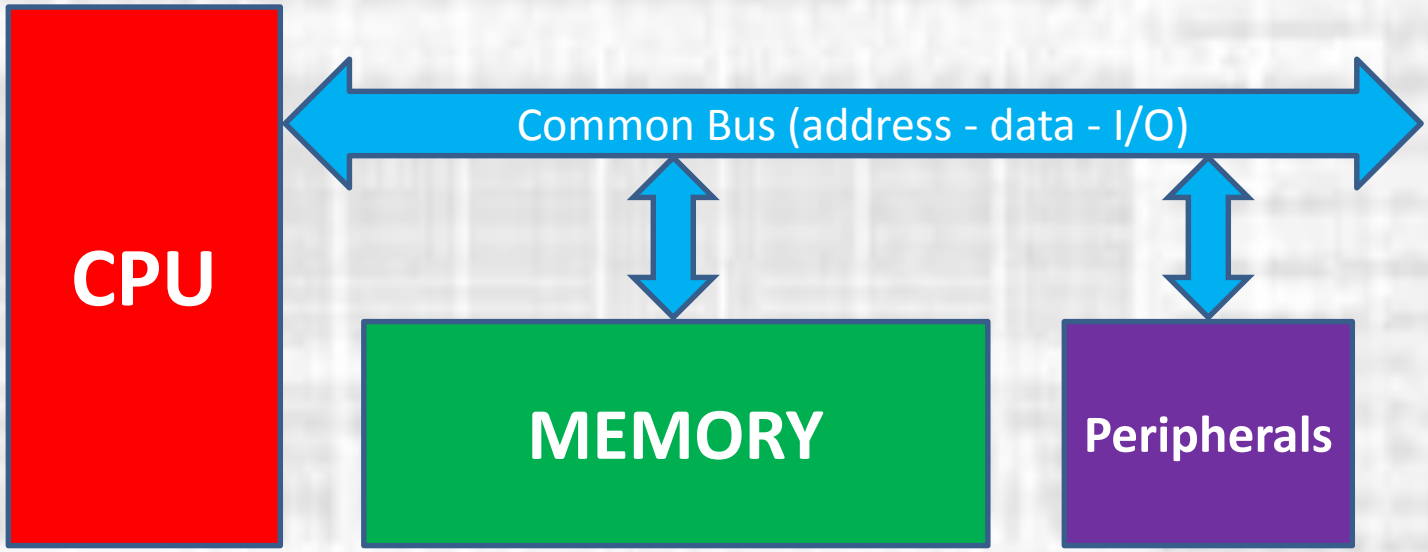
- Modern Example



Program Execution

Simplified Execution

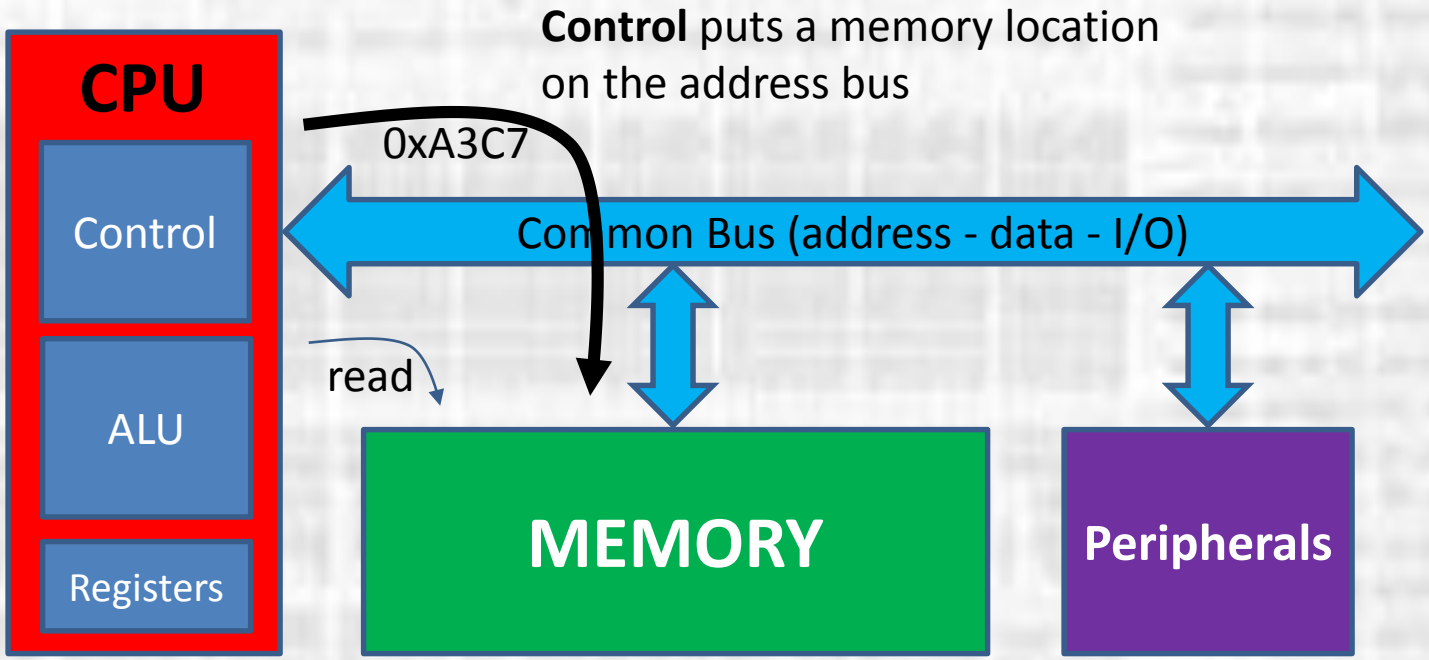
- Simplified Block Diagram



Program Execution

Simplified Execution

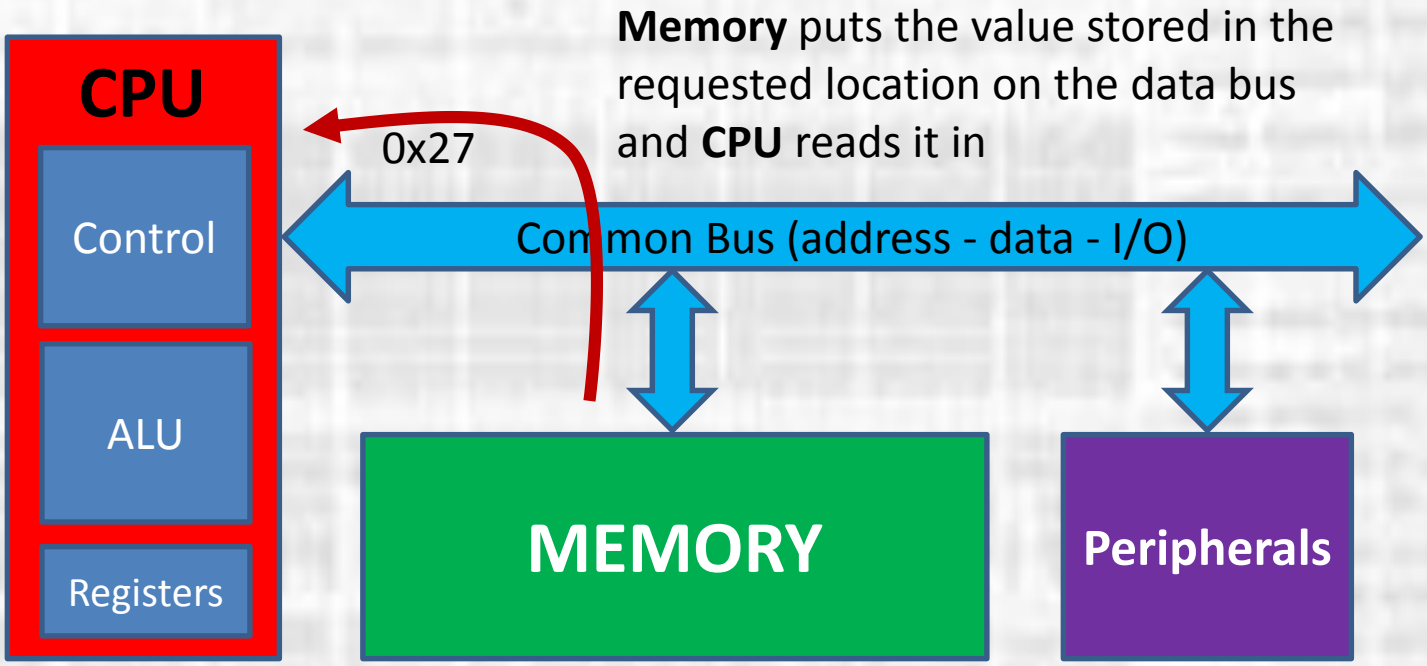
- Request Instruction (fetch)



Program Execution

Simplified Execution

- Request Instruction (fetch)

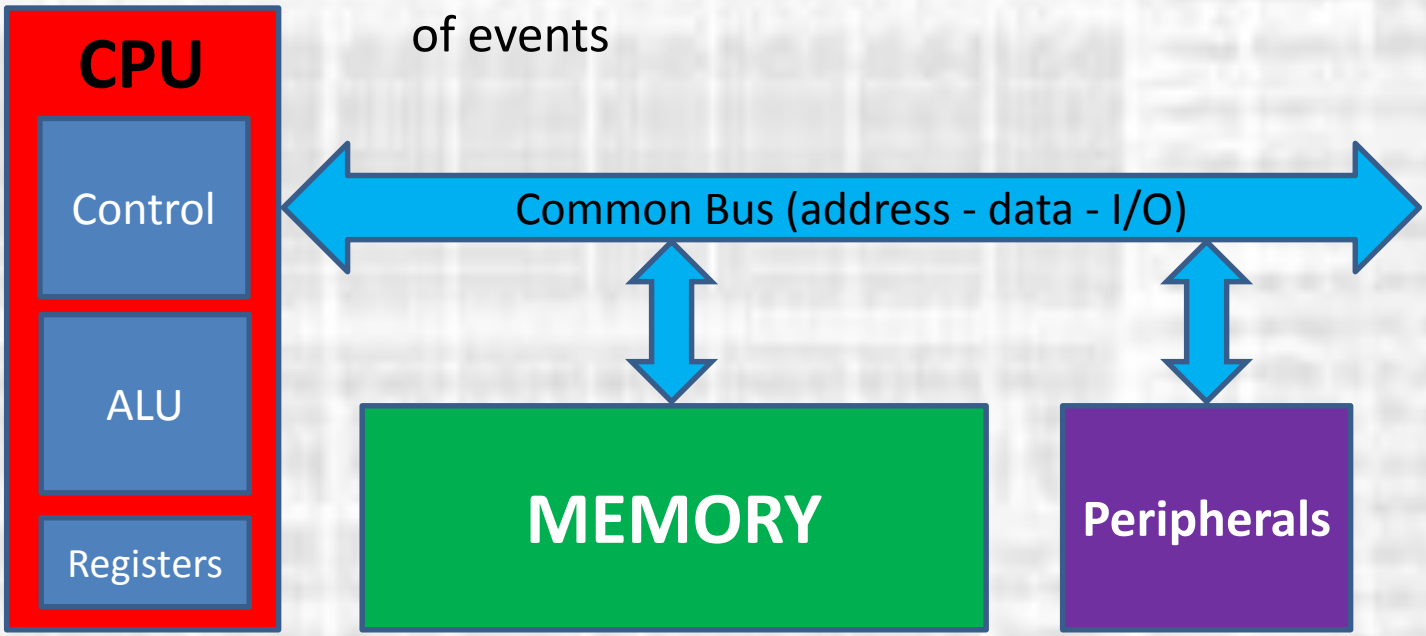


Program Execution

Simplified Execution

- Decode Instruction (decode)

Control decodes the byte returned by the memory and prepares to execute a pre-defined sequence of events

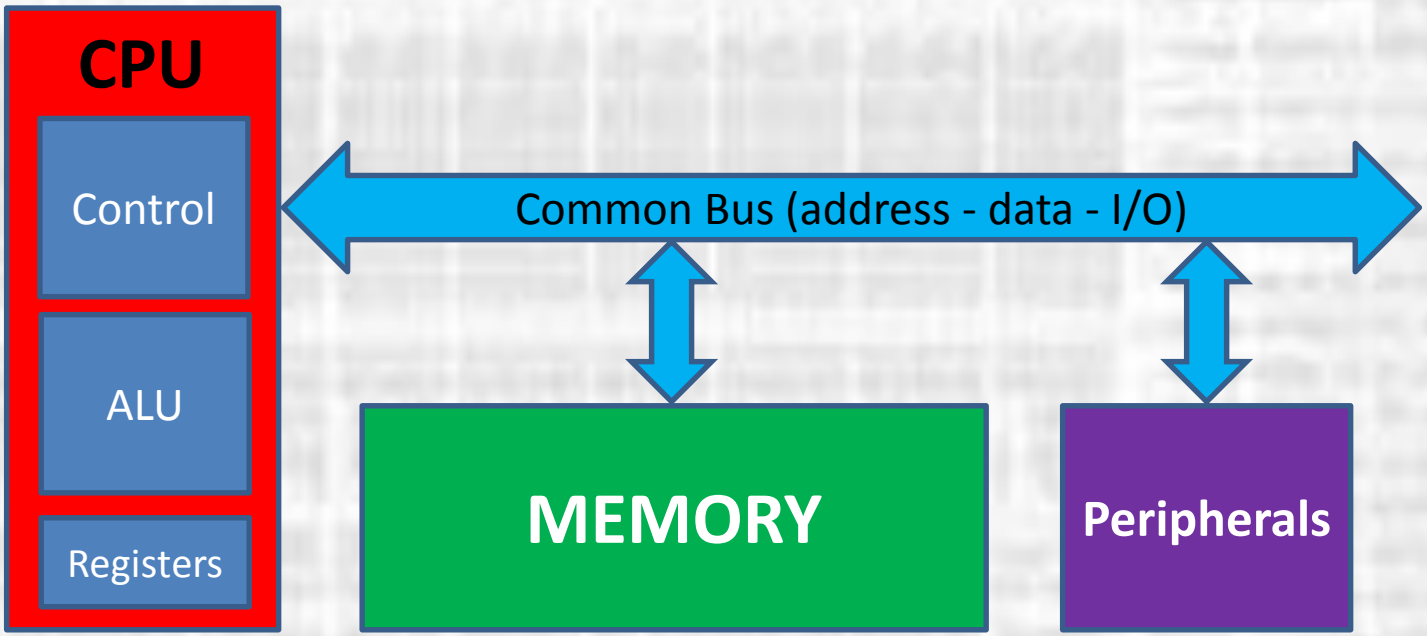


Program Execution

Simplified Execution

- Execute Instruction (execute)

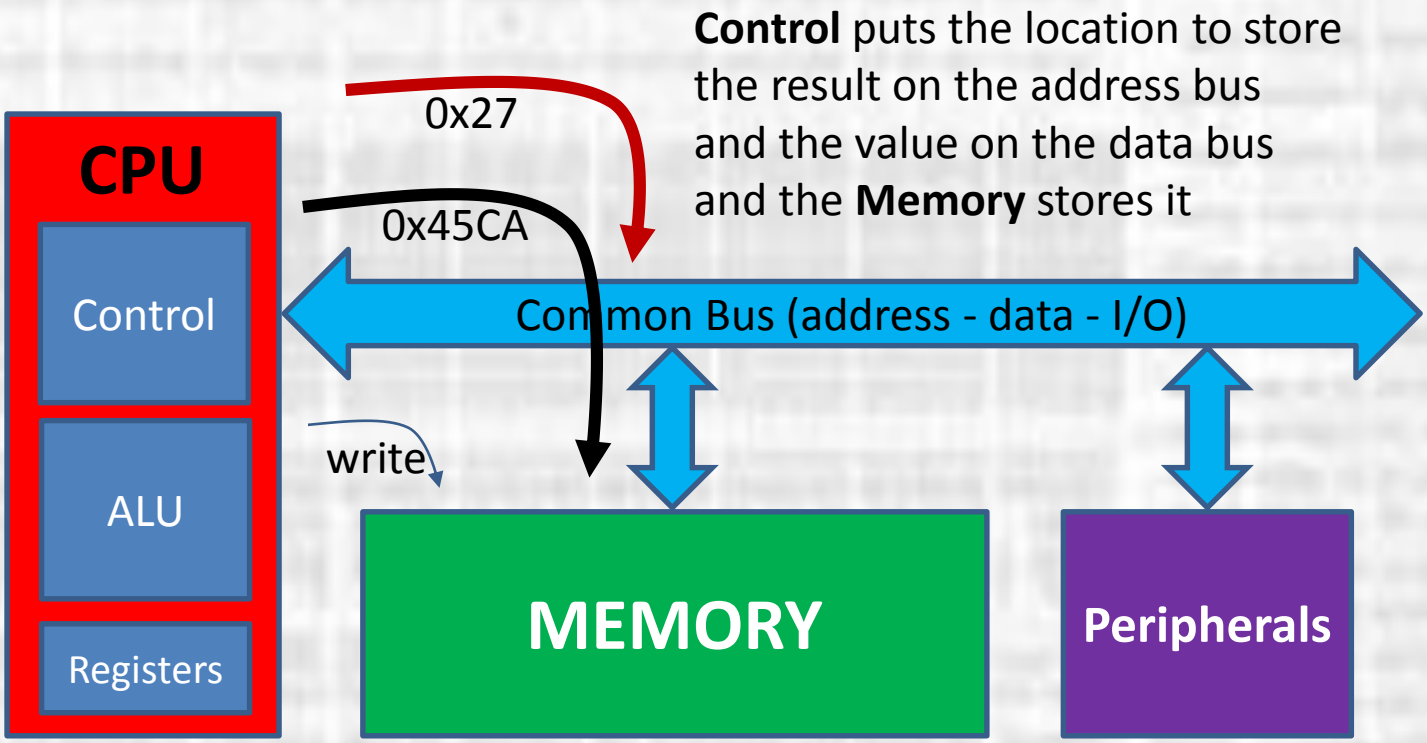
CPU executes the fetched instruction (pre-defined sequence of events)



Program Execution

Simplified Execution

- Write Result (writeback)



Program Execution

Program Basics

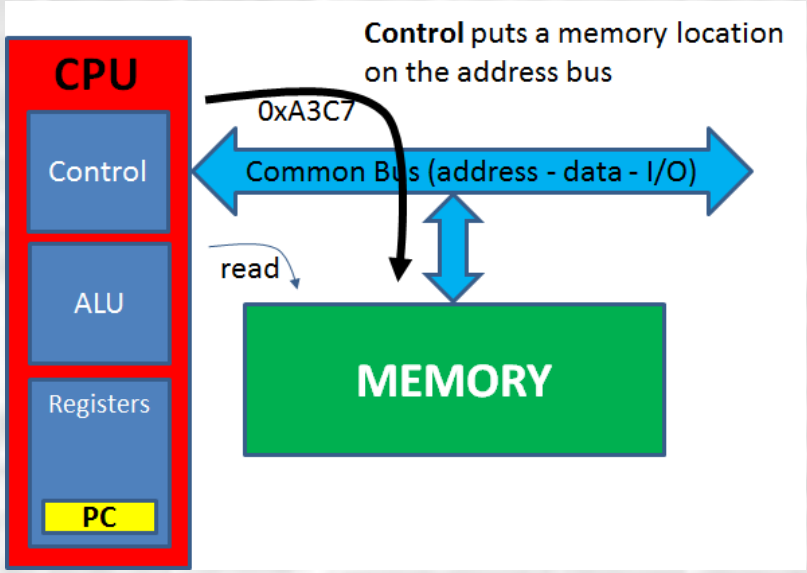
- Memory Structure
  - Each memory location (address) holds 1 byte of information (data)
  - Data can represent instructions, numbers, address offsets, …
  - Typically represent memory as expanding downward
  - # of address bits determines the maximum memory size
  - All addresses are not necessarily tied to the same memory device – or to memory at all
  - Most memories include hidden bits to increase reliability (parity)

ADDRESS	DATA
Binary	Binary
0000 0000 0000 0000	0111 0101
0000 0000 0000 0001	0010 0000
0000 0000 0000 0010	0000 0000
0000 0000 0000 0011	0111 0101
0000 0000 0000 0100	0010 0001
0000 0000 0000 0101	0001 0100
0000 0000 0000 0110	1001 0000
0000 0000 0000 0111	0000 0000
0000 0000 0111 0011	0111 0101
0000 0000 0111 0100	0010 0000
0000 0000 0111 0101	0000 0000
0000 0000 0111 0110	0111 0101
0000 0000 0111 0111	0010 0001
0000 0000 0111 1000	0001 0100
0000 0000 0111 1010	1001 0000
0000 0000 0111 1011	0000 0000
1111 1111 1111 1000	0111 0101
1111 1111 1111 1001	0010 0000
1111 1111 1111 1010	0000 0000
1111 1111 1111 1011	0111 0101
1111 1111 1111 1100	0010 0001
1111 1111 1111 1101	0001 0100
1111 1111 1111 1110	1001 0000
1111 1111 1111 1111	0000 0000

Program Execution

Program Basics

- Program Counter (PC)
  - The PC holds the location (address) of the **NEXT** instruction to fetch (execute)
  - At power-up or after a reset the PC is forced into a known state – typically 0x0000 or 0xFFFE
  - Normal program flow: PC is incremented by 1 after each memory read
  - Branches – if taken, the branch offset is added to the PC → PC
  - Jumps – PC is loaded with the jump target



Program Execution

Program Basics

- Program Counter (PC)
  - The PC holds the location (address) of the **NEXT** instruction to fetch (execute)
  - At power-up or after a reset the PC is forced into a known state – typically 0x0000 or 0xFFFE
  - Normal program flow: PC is incremented by 1 after each memory read
  - Branches – if taken, the branch offset is added to the PC → PC
  - Jumps – PC is loaded with the jump target

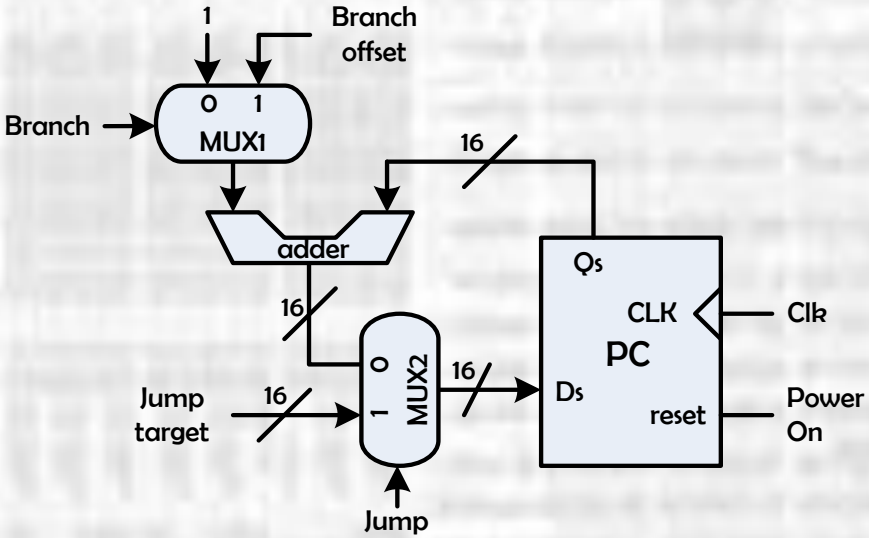


Figure 1.4 A simplified block diagram of the program counter (PC) of an 8-bit microcontroller

src: Huang

Program Execution

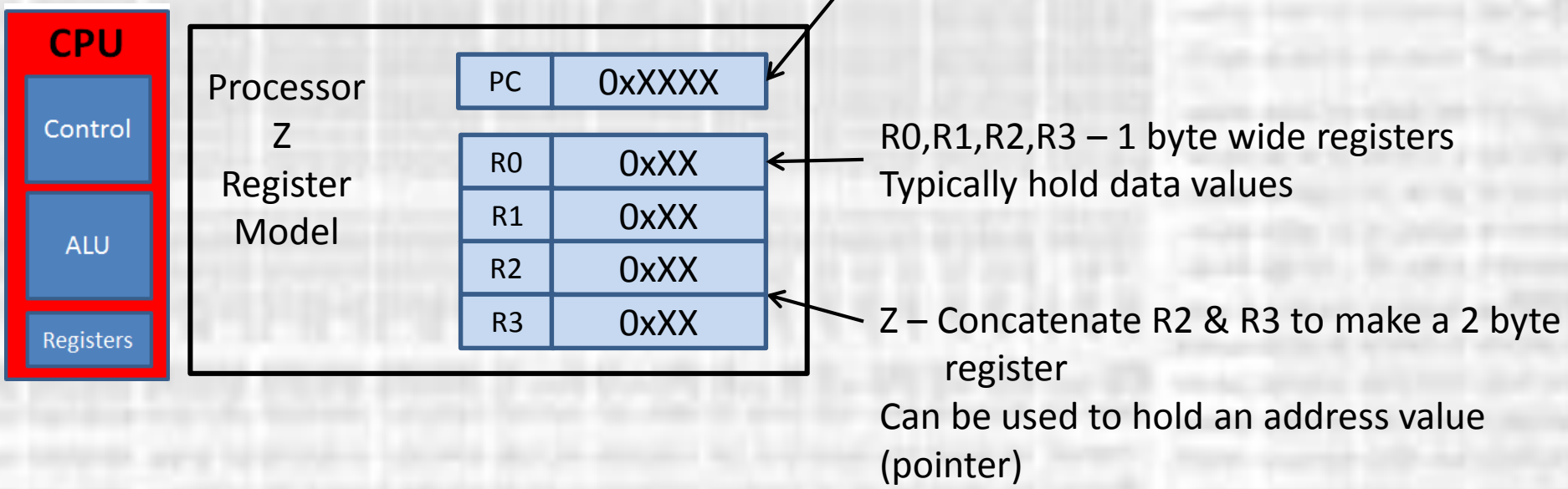
Program Basics

- C / Assembly / Machine Languages
  - C Language
    - high level – highly abstracted
    - Transportable – not machine / processor dependent
    - Compiler/assembler converts C into machine specific assembly language
    - Readable by a wide array of programmers
  - Assembly language
    - Low level - machine specific
    - Refers to the actual machine instructions and registers
    - Readable by knowledgeable programmers
  - Machine language
    - Very low level – tied to a specific processor implementation
    - 1's and 0's
    - Generally not readable without a lot of effort
    - This is what is actually stored in the memory and executed by the processor

Program Execution

Processor Z

- Register Model
  - A description of the available registers to be used by the processor



Program Execution

Processor Z

- Execution Model
  - Operation is broken into 3 parts
  - All 3 parts occur for each clock, but are distinct operations

A) Fetch and Increment PC
   Fetch the instruction pointed to by the current PC
   Increment the PC after fetch is complete

B) Decode
   Decode the instruction

C) Execute
   Perform any arithmetic operation
   or
   Perform a Load or Store to memory

Program Execution

Processor Z

- Instruction Set

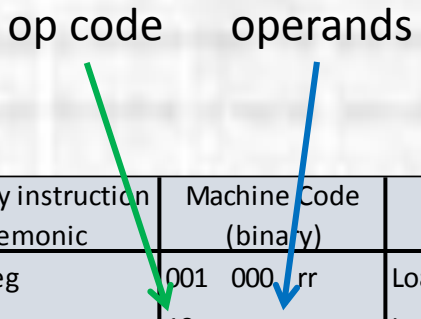
mnemonic operands

Assembly instruction mnemonic		Machine Code (binary)	Action
ldz	reg		Load register from memory location pointed to by register Z
ldi	reg, v		Load immediate register with value v
stz	reg		Store register to memory location pointed to by register Z
add	regd, regs		Add regd to regs and place result in regd (regd ← regd + regs)
dec	reg		Decrement the contents of register
bnz	reg, offset		If the value in reg is not zero, branch to the location given by PC + offset
Notes:			

Program Execution

Processor Z

- Instruction Set



Assembly instruction mnemonic		Machine Code (binary)	Action
ldz	reg	001 000 rr	Load register from memory location pointed to by register Z
ldi	reg, vv	10 vvvv rr	Load immediate register with value v
stz	reg	011 000 rr	Store register to memory location pointed to by register Z
add	regd, regs	000 0 rd rs	Add regd to regs and place result in regd (regd ← regd + regs)
dec	reg	010 000 rr	Decrement the contents of register
bnz	reg, offset	11 vvvv rr	If the value in reg is not zero, branch to the location given by PC + offset (vvvv)

Notes: rr : a 2-bit value that represents register R0, R1, R2, R3
rs,rd : a 2-bit value that represents register R0, R1, R2, R3
vvvv: a 4 bit value representing a scalar value in 2's comliment

- Each op code must be unique – the processor must determine what to do based on the op code.

Program Execution

Processor Z

- Instruction Format

Label	Operation	Operands	Comments
loop:	add	R1,R0	; R1 = R1 + R0

Label:

not stored in the computer

used by the compiler to keep track of where things are located in memory

indicates the memory location of the Operation (instruction) immediately following the label

Comments;

All good code should be properly commented

Program Execution

Processor Z

- Instruction Format
  - Formats
    - Capitalization does not matter
    - Spaces between operands do not matter
    - Unmarked values are considered decimal
      - Hex: 0xXX, XXh
      - Bin: XXb, bXX
      - Dec: XX

```
LDI  R1, 5
ldi          r1, 0x05
Ldi  R1,     0x5
lDi  r1, 00000101b
```

Program Execution

Processor Z

- Instruction Format
  - Numerical values
    - All addresses are considered unsigned binary
      - No negative memory addresses
    - All numerical values are considered 2's compliment binary
      - Includes loop offsets

Program Execution

Assembly Program

- Assembly Language Program
  - Calculate sum of 1 to N

Input:
N is located in memory location 0x0402
Output:
sum is to be placed in memory location 0x0403

Program Strategy:
R0 – count
R1 – sum
Set R0 equal to N, add to sum, loop until count (R0) is zero

Program Execution

Assembly Program

- Assembly Language Program

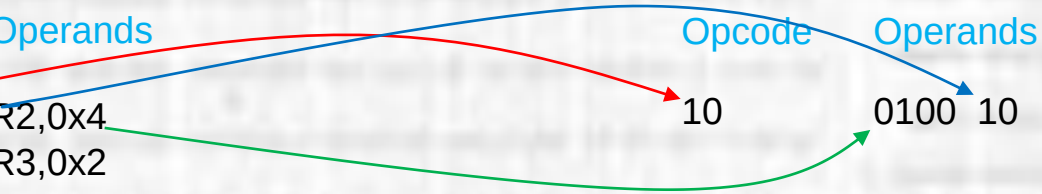
```
Label   Operation   Operands    Comments

        ldi         R2,0x4      ; place upper half of address in R2
        ldi         R3,0x2      ; place lower half of address in R3
                                ; Z now holds 0x0402 (location of N)
        ldz         R0          ; load R0 "N" from memory location 0x0402
        ldi         R1,0x0      ; clear R1 (sum)

loop:   add         R1,R0       ; R1 = R1 + R0
        dec         R0          ; decrement "N"
        bnz         R0,loop     ; branch to "loop" if R0 is not 0

        ldi         R3,0x3      ; replace lower half of memory location for Z
        stz         R1          ; store the sum at memory location pointed to
                                ; by Z (0x0403)
```

Program Execution

Assembly Program

- Assembly Language Program

Label	Operation	Operands
	ldi	R2,0x4
	ldi	R3,0x2
	ldz	R0
	ldi	R1,0x0
loop:	add	R1,R0
	dec	R0
	bnz	R0,loop
	ldi	R3,0x3
	stz	R1



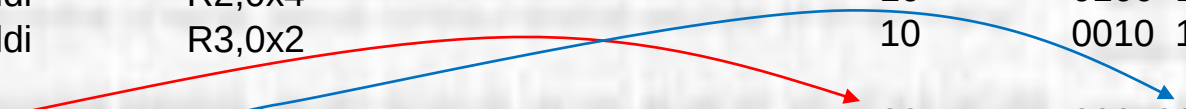
Assembly instruction mnemonic		Machine Code (binary)	Action
ldz	reg	001 000 rr	Load register from memory location pointed to by register Z
ldi	reg, vv	10 vvvv rr	Load immediate register with value v
stz	reg	011 000 rr	Store register to memory location pointed to by register Z
add	regd, regs	000 0 rd rs	Add regd to regs and place result in regd (regd ← regd + regs)
dec	reg	010 000 rr	Decrement the contents of register
bnz	reg, offset	11 vvvv rr	If the value in reg is not zero, branch to the location given by PC + offset (vvvv)

Program Execution

Assembly Program

- Assembly Language Program

Label	Operation	Operands	Opcode	Operands
	ldi	R2,0x4	10	0100 10
	ldi	R3,0x2	10	0010 11
	ldz	R0	001	000 00
	ldi	R1,0x0		
loop:	add	R1,R0		
	dec	R0		
	bnz	R0,loop		
	ldi	R3,0x3		
	stz	R1		



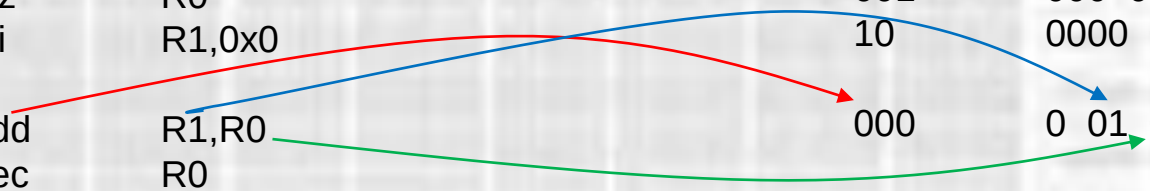
Assembly instruction mnemonic		Machine Code (binary)	Action
ldz	reg	001 000 rr	Load register from memory location pointed to by register Z
ldi	reg, vv	10 vvvv rr	Load immediate register with value v
stz	reg	011 000 rr	Store register to memory location pointed to by register Z
add	regd, regs	000 0 rd rs	Add regd to regs and place result in regd (regd ← regd + regs)
dec	reg	010 000 rr	Decrement the contents of register
bnz	reg, offset	11 vvvv rr	If the value in reg is not zero, branch to the location given by PC + offset (vvvv)

Program Execution

Assembly Program

- Assembly Language Program

Label	Operation	Operands	Opcode	Operands
	ldi	R2,0x4	10	0100 10
	ldi	R3,0x2	10	0010 11
	ldz	R0	001	000 00
	ldi	R1,0x0	10	0000 01
loop:	add	R1,R0	000	0 01 00
	dec	R0		
	bnz	R0,loop		
	ldi	R3,0x3		
	stz	R1		



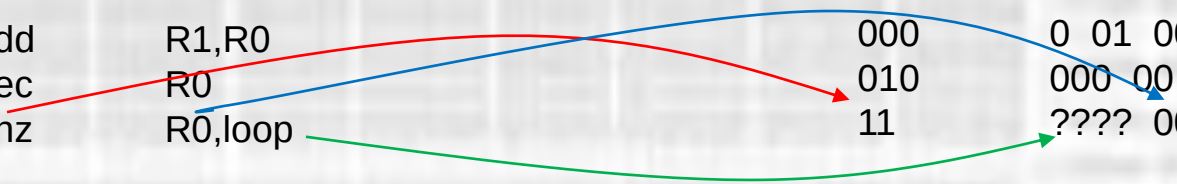
Assembly instruction mnemonic		Machine Code (binary)	Action
ldz	reg	001 000 rr	Load register from memory location pointed to by register Z
ldi	reg, vv	10 vvvv rr	Load immediate register with value v
stz	reg	011 000 rr	Store register to memory location pointed to by register Z
add	regd, regs	000 0 rd rs	Add regd to regs and place result in regd (regd ← regd + regs)
dec	reg	010 000 rr	Decrement the contents of register
bnz	reg, offset	11 vvvv rr	If the value in reg is not zero, branch to the location given by PC + offset (vvvv)

Program Execution

Assembly Program

- Assembly Language Program

Label	Operation	Operands	Opcode	Operands
	ldi	R2,0x4	10	0100 10
	ldi	R3,0x2	10	0010 11
	ldz	R0	001	000 00
	ldi	R1,0x0	10	0000 01
loop:	add	R1,R0	000	0 01 00
	dec	R0	010	000 00
	bnz	R0,loop	11	???? 00
	ldi	R3,0x3		
	stz	R1		



Assembly instruction mnemonic		Machine Code (binary)	Action
ldz	reg	001 000 rr	Load register from memory location pointed to by register Z
ldi	reg, vv	10 vvvv rr	Load immediate register with value v
stz	reg	011 000 rr	Store register to memory location pointed to by register Z
add	regd, regs	000 0 rd rs	Add regd to regs and place result in regd (regd ← regd + regs)
dec	reg	010 000 rr	Decrement the contents of register
bnz	reg, offset	11 vvvv rr	If the value in reg is not zero, branch to the location given by PC + offset (vvvv)

Program Execution

Assembly Program

- Assembly to Machine Language

Label	Operation	Operands	Opcode	Operands
	ldi	R2,0x4	10	0100 10
	ldi	R3,0x2	10	0010 11
	ldz	R0	001	000 00
	ldi	R1,0x0	10	0000 01
loop:	add	R1,R0	000	0 01 00
	dec	R0	010	000 00
	bnz	R0,loop	11	???? 00
	ldi	R3,0x3	10	0011 11
	stz	R1	011	000 01

Assembly instruction mnemonic		Machine Code (binary)	Action
ldz	reg	001 000 rr	Load register from memory location pointed to by register Z
ldi	reg, vv	10 vvvv rr	Load immediate register with value v
stz	reg	011 000 rr	Store register to memory location pointed to by register Z
add	regd, regs	000 0 rd rs	Add regd to regs and place result in regd (regd ← regd + regs)
dec	reg	010 000 rr	Decrement the contents of register
bnz	reg, offset	11 vvvv rr	If the value in reg is not zero, branch to the location given by PC + offset (vvvv)

Program Execution

Assembly Program

- Machine Language Program

Label	Operation	Operands	Opcode	Operands
	ldi	R2,0x4	10	0100 10
	ldi	R3,0x2	10	0010 11
	ldz	R0	001	000 00
	ldi	R1,0x0	10	0000 01
loop:	add	R1,R0	000	0 01 00
	dec	R0	010	000 00
	bnz	R0,loop	11	???? 00
	ldi	R3,0x3	10	0011 11
	stz	R1	011	000 01

ADDRESS				DATA	
Hex	Binary			Binary	Hex
0000	0000 0000 0000 0000			1001 0010	92
0001	0000 0000 0000 0001			1000 1011	8B
0002	0000 0000 0000 0010			0010 0000	20
0003	0000 0000 0000 0011			1000 0001	81
0004	0000 0000 0000 0100			0000 0100	04
0005	0000 0000 0000 0101			0100 0000	40
0006	0000 0000 0000 0110			11?? ??00	F4
0007	0000 0000 0000 0111			1000 1111	8F
0008	0000 0000 0000 1000			0110 0001	61

Program Execution

Assembly Program

- Assembly Language Program

Label	Operation	Operands	Opcode	Operands
	ldi	R2,0x4	10	0100 10
	ldi	R3,0x2	10	0010 11
	ldz	R0	001	000 00
	ldi	R1,0x0	10	0000 01
loop:	add	R1,R0	000	0 01 00
	dec	R0	010	000 00
	bnz	R0,loop	11	???? 00
	ldi	R3,0x3	10	0011 11
	stz	R1	011	000 01

ADDRESS		DATA	
Hex	Binary	Binary	Hex
0000	0000 0000 0000 0000	1001 0010	92
0001	0000 0000 0000 0001	1000 1011	8B
0002	0000 0000 0000 0010	0010 0000	20
0003	0000 0000 0000 0011	1000 0001	81
0004	0000 0000 0000 0100	0000 0100	04
0005	0000 0000 0000 0101	0100 0000	40
0006	0000 0000 0000 0110	1111 0100	F4
0007	0000 0000 0000 0111	1000 1111	8F
0008	0000 0000 0000 1000	0110 0001	61

Loop back 2 instructions – BUT – PC has already incremented

Loop back 3 → -3 → 1101 in 2's compliment

Program Execution

Processor Z

ADDRESS																DATA									
Hex				Binary												Binary								Hex	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	1	0	9	2
0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	1	0	1	1	8	B
0	0	0	2	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	2	0
0	0	0	3	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	1	8	1
0	0	0	4	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	4
0	0	0	5	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0	0	0	0	0	0	4	0
0	0	0	6	0	0	0	0	0	0	0	0	0	1	1	0	1	1	1	1	0	1	0	0	F	4
0	0	0	7	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	1	1	1	1	8	F
0	0	0	8	0	0	0	0	0	0	0	0	1	0	0	0	0	1	1	0	0	0	0	1	6	1
0	4	0	0	0	0	0	0	0	1	0	0	0	0	0	0	x	x	x	x	x	x	x	x	x	x
0	4	0	1	0	0	0	0	0	1	0	0	0	0	0	1	x	x	x	x	x	x	x	x	x	x
0	4	0	2	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	1	1	0	3
0	4	0	3	0	0	0	0	0	1	0	0	0	0	1	1	x	x	x	x	x	x	x	x	x	x

Initial Memory Contents

PC	0xXXXX

R0	0xXX
R1	0xXX

R2	0xXX
R3	0xXX

Z

Fetched data	
Decoded Instruction	

Processor state unknown

Program Execution

Processor Z

ADDRESS																DATA									
Hex				Binary												Binary								Hex	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	1	0	9	2
0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	1	0	1	1	8	B
0	0	0	2	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	2	0
0	0	0	3	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	1	8	1
0	0	0	4	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	4
0	0	0	5	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0	0	0	0	0	0	4	0
0	0	0	6	0	0	0	0	0	0	0	0	0	1	1	0	1	1	1	1	0	1	0	0	F	4
0	0	0	7	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	1	1	1	1	8	F
0	0	0	8	0	0	0	0	0	0	0	0	1	0	0	0	0	1	1	0	0	0	0	1	6	1
0	4	0	0	0	0	0	0	0	1	0	0	0	0	0	0	x	x	x	x	x	x	x	x	x	x
0	4	0	1	0	0	0	0	0	1	0	0	0	0	0	1	x	x	x	x	x	x	x	x	x	x
0	4	0	2	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	1	1	0	3
0	4	0	3	0	0	0	0	0	1	0	0	0	0	1	1	x	x	x	x	x	x	x	x	x	x

Perform Reset

PC	0x0000

R0	0xXX
R1	0xXX

R2	0xXX
R3	0xXX

Z

Fetched data	
Decoded Instruction	

PC is loaded with 0x000

Program Execution

Processor Z

ADDRESS																DATA									
Hex				Binary												Binary								Hex	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	1	0	9	2
0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	1	0	1	1	8	B
0	0	0	2	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	2	0
0	0	0	3	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	1	8	1
0	0	0	4	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	4
0	0	0	5	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0	0	0	0	0	0	4	0
0	0	0	6	0	0	0	0	0	0	0	0	0	1	1	0	1	1	1	1	0	1	0	0	F	4
0	0	0	7	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	1	1	1	1	8	F
0	0	0	8	0	0	0	0	0	0	0	0	1	0	0	0	0	1	1	0	0	0	0	1	6	1
0	4	0	0	0	0	0	0	0	1	0	0	0	0	0	0	x	x	x	x	x	x	x	x	x	x
0	4	0	1	0	0	0	0	0	1	0	0	0	0	0	1	x	x	x	x	x	x	x	x	x	x
0	4	0	2	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	1	1	0	3
0	4	0	3	0	0	0	0	0	1	0	0	0	0	1	1	x	x	x	x	x	x	x	x	x	x

Perform Reset

PC	0x0001

R0	0xXX
R1	0xXX

R2	0xXX
R3	0xXX

Z

Fetched data	0x92
Decoded Instruction	

Clk 1

Fetch and increment PC

Program Execution

Processor Z

ADDRESS																	DATA									
Hex				Binary													Binary								Hex	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	1	0	9	2
0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	1	0	1	1	8	B
0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	2	0
0	0	0	3	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	1	8	1
0	0	0	4	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	4
0	0	0	5	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0	0	0	0	0	4	0
0	0	0	6	0	0	0	0	0	0	0	0	0	0	0	1	1	0	1	1	1	1	0	1	0	F	4
0	0	0	7	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	1	1	1	8	F
0	0	0	8	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	1	0	0	0	0	6	1
0	4	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	x	x	x	x	x	x	x	x	x
0	4	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	1	x	x	x	x	x	x	x	x	x
0	4	0	2	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	1	1	0	3
0	4	0	3	0	0	0	0	0	1	0	0	0	0	0	0	1	1	x	x	x	x	x	x	x	x	x

PC	0x0001

R0	0xXX
R1	0xXX

R2	0xXX
R3	0xXX

Z

Fetched data	0x92
Decoded Instruction	
10 0100 10	
Ldi R2, 4	

Clk 1
Decode

Assembly instruction mnemonic		Machine Code (binary)	Action
ldz	reg	001 000 rr	Load register from memory location pointed to by register Z
ldi	reg, vv	10 vvvv rr	Load immediate register with value v
stz	reg	011 000 rr	Store register to memory location pointed to by register Z
add	regd, regs	000 0 rd rs	Add regd to regs and place result in regd (regd ← regd + regs)
dec	reg	010 000 rr	Decrement the contents of register
bnz	reg, offset	11 vvvv rr	If the value in reg is not zero, branch to the location given by PC + offset (vvvv)

Program Execution

Processor Z

ADDRESS																	DATA									
Hex				Binary													Binary								Hex	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	1	0	9	2
0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	1	0	1	1	8	B
0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	2	0
0	0	0	3	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	1	8	1
0	0	0	4	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	4
0	0	0	5	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0	0	0	0	0	4	0
0	0	0	6	0	0	0	0	0	0	0	0	0	0	0	1	1	0	1	1	1	1	0	1	0	F	4
0	0	0	7	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	1	1	1	8	F
0	0	0	8	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	1	0	0	0	0	6	1
0	4	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	x	x	x	x	x	x	x	x	x
0	4	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	1	x	x	x	x	x	x	x	x	x
0	4	0	2	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	1	1	0	3
0	4	0	3	0	0	0	0	0	1	0	0	0	0	0	0	1	1	x	x	x	x	x	x	x	x	x

PC	0x0001

R0	0xXX
R1	0xXX

R2	0x04
R3	0xXX

Z

Fetched data	0x92
Decoded Instruction	
10 0100 10	
Ldi R2, 4	

Clk 1
Execute

Program Execution

Processor Z

ADDRESS																DATA									
Hex				Binary												Binary								Hex	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	1	0	9	2
0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	1	0	1	1	8	B
0	0	0	2	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	2	0
0	0	0	3	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	1	8	1
0	0	0	4	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	4
0	0	0	5	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0	0	0	0	0	0	4	0
0	0	0	6	0	0	0	0	0	0	0	0	0	1	1	0	1	1	1	1	0	1	0	0	F	4
0	0	0	7	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	1	1	1	1	8	F
0	0	0	8	0	0	0	0	0	0	0	0	1	0	0	0	0	1	1	0	0	0	0	1	6	1
0	4	0	0	0	0	0	0	0	1	0	0	0	0	0	0	x	x	x	x	x	x	x	x	x	x
0	4	0	1	0	0	0	0	0	1	0	0	0	0	0	1	x	x	x	x	x	x	x	x	x	x
0	4	0	2	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	1	1	0	3
0	4	0	3	0	0	0	0	0	1	0	0	0	0	1	1	x	x	x	x	x	x	x	x	x	x

PC	0x0002

R0	0xXX
R1	0xXX

R2	0x04
R3	0xXX

Z

Fetched data	0x8B
Decoded Instruction	

Clk 2

Fetch and Increment PC

Program Execution

Processor Z

ADDRESS																DATA									
Hex				Binary												Binary								Hex	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	1	0	9	2
0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	1	0	1	1	8	B
0	0	0	2	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	2	0
0	0	0	3	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	1	8	1
0	0	0	4	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	4
0	0	0	5	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0	0	0	0	0	0	4	0
0	0	0	6	0	0	0	0	0	0	0	0	0	1	1	0	1	1	1	1	0	1	0	0	F	4
0	0	0	7	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	1	1	1	1	8	F
0	0	0	8	0	0	0	0	0	0	0	0	1	0	0	0	0	1	1	0	0	0	0	1	6	1
0	4	0	0	0	0	0	0	0	1	0	0	0	0	0	0	x	x	x	x	x	x	x	x	x	x
0	4	0	1	0	0	0	0	0	1	0	0	0	0	0	1	x	x	x	x	x	x	x	x	x	x
0	4	0	2	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	1	1	0	3
0	4	0	3	0	0	0	0	0	1	0	0	0	0	1	1	x	x	x	x	x	x	x	x	x	x

PC	0x0002

R0	0xXX
R1	0xXX

R2	0x04
R3	0xXX

Z

Fetched data	0x8B
Decoded Instruction	
10 0010 11	
Ldi R3, 2	

Clk 2
Decode

Assembly instruction mnemonic		Machine Code (binary)	Action
ldz	reg	001 000 rr	Load register from memory location pointed to by register Z
ldi	reg, vv	10 vvvv rr	Load immediate register with value v
stz	reg	011 000 rr	Store register to memory location pointed to by register Z
add	regd, regs	000 0 rd rs	Add regd to regs and place result in regd (regd ← regd + regs)
dec	reg	010 000 rr	Decrement the contents of register
bnz	reg, offset	11 vvvv rr	If the value in reg is not zero, branch to the location given by PC + offset (vvvv)

Program Execution

Processor Z

ADDRESS																	DATA									
Hex				Binary													Binary								Hex	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	1	0	9	2
0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	1	0	1	1	8	B
0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	2	0
0	0	0	3	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	1	8	1
0	0	0	4	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	4
0	0	0	5	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0	0	0	0	0	4	0
0	0	0	6	0	0	0	0	0	0	0	0	0	0	0	1	1	0	1	1	1	1	0	1	0	F	4
0	0	0	7	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	1	1	1	8	F
0	0	0	8	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	1	0	0	0	0	6	1
0	4	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	x	x	x	x	x	x	x	x	x
0	4	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	1	x	x	x	x	x	x	x	x	x
0	4	0	2	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	1	1	0	3
0	4	0	3	0	0	0	0	0	1	0	0	0	0	0	0	1	1	x	x	x	x	x	x	x	x	x

PC	0x0002

R0	0xXX
R1	0xXX

R2	0x04
R3	0x02

Z

Fetched data	0x8B
Decoded Instruction	
10 0010 11	
Ldi R3, 2	

Clk 2
Execute

Program Execution

Processor Z

ADDRESS																DATA									
Hex				Binary												Binary								Hex	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	1	0	9	2
0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	1	0	1	1	8	B
0	0	0	2	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	2	0
0	0	0	3	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	1	8	1
0	0	0	4	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	4
0	0	0	5	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0	0	0	0	0	0	4	0
0	0	0	6	0	0	0	0	0	0	0	0	0	1	1	0	1	1	1	1	0	1	0	0	F	4
0	0	0	7	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	1	1	1	1	8	F
0	0	0	8	0	0	0	0	0	0	0	0	1	0	0	0	0	1	1	0	0	0	0	1	6	1
0	4	0	0	0	0	0	0	0	1	0	0	0	0	0	0	x	x	x	x	x	x	x	x	x	x
0	4	0	1	0	0	0	0	0	1	0	0	0	0	0	1	x	x	x	x	x	x	x	x	x	x
0	4	0	2	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	1	1	0	3
0	4	0	3	0	0	0	0	0	1	0	0	0	0	1	1	x	x	x	x	x	x	x	x	x	x

PC	0x0003

R0	0xXX
R1	0xXX

R2	0x04
R3	0x02

Z

Fetched data	0x20
Decoded Instruction	

Clk 3

Fetch and increment PC

Program Execution

Processor Z

ADDRESS																DATA									
Hex				Binary												Binary								Hex	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	1	0	9	2
0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	1	0	1	1	8	B
0	0	0	2	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	2	0
0	0	0	3	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	1	8	1
0	0	0	4	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	4
0	0	0	5	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0	0	0	0	0	0	4	0
0	0	0	6	0	0	0	0	0	0	0	0	0	1	1	0	1	1	1	1	0	1	0	0	F	4
0	0	0	7	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	1	1	1	1	8	F
0	0	0	8	0	0	0	0	0	0	0	0	1	0	0	0	0	1	1	0	0	0	0	1	6	1
0	4	0	0	0	0	0	0	0	1	0	0	0	0	0	0	x	x	x	x	x	x	x	x	x	x
0	4	0	1	0	0	0	0	0	1	0	0	0	0	0	1	x	x	x	x	x	x	x	x	x	x
0	4	0	2	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	1	1	0	3
0	4	0	3	0	0	0	0	0	1	0	0	0	0	1	1	x	x	x	x	x	x	x	x	x	x

PC	0x0003

R0	0xXX
R1	0xXX

R2	0x04
R3	0x02

Z

Fetched data	0x20
Decoded Instruction	
001 000 00	
Ldz R0	

Clk 3
Decode

Assembly instruction mnemonic		Machine Code (binary)	Action
ldz	reg	001 000 rr	Load register from memory location pointed to by register Z
ldi	reg, vv	10 vvvv rr	Load immediate register with value v
stz	reg	011 000 rr	Store register to memory location pointed to by register Z
add	regd, regs	000 0 rd rs	Add regd to regs and place result in regd (regd ← regd + regs)
dec	reg	010 000 rr	Decrement the contents of register
bnz	reg, offset	11 vvvv rr	If the value in reg is not zero, branch to the location given by PC + offset (vvvv)

Program Execution

Processor Z

ADDRESS																DATA									
Hex				Binary												Binary								Hex	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	1	0	9	2
0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	1	0	1	1	8	B
0	0	0	2	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	2	0
0	0	0	3	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	1	8	1
0	0	0	4	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	4
0	0	0	5	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0	0	0	0	0	0	4	0
0	0	0	6	0	0	0	0	0	0	0	0	0	1	1	0	1	1	1	1	0	1	0	0	F	4
0	0	0	7	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	1	1	1	1	8	F
0	0	0	8	0	0	0	0	0	0	0	0	1	0	0	0	0	1	1	0	0	0	0	1	6	1
0	4	0	0	0	0	0	0	0	1	0	0	0	0	0	0	x	x	x	x	x	x	x	x	x	x
0	4	0	1	0	0	0	0	0	1	0	0	0	0	0	1	x	x	x	x	x	x	x	x	x	x
0	4	0	2	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	1	1	0	3
0	4	0	3	0	0	0	0	0	1	0	0	0	0	1	1	x	x	x	x	x	x	x	x	x	x

PC	0x0003

R0	0x03
R1	0xXX

R2	0x04
R3	0x02

Z

Fetched data	0x20
Decoded Instruction	
001 000 00	
Ldz R0	

Clk 3
Execute

Program Execution

Processor Z

ADDRESS																DATA									
Hex				Binary												Binary								Hex	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	1	0	9	2
0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	1	0	1	1	8	B
0	0	0	2	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	2	0
0	0	0	3	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	1	8	1
0	0	0	4	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	4
0	0	0	5	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0	0	0	0	0	0	4	0
0	0	0	6	0	0	0	0	0	0	0	0	0	1	1	0	1	1	1	1	0	1	0	0	F	4
0	0	0	7	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	1	1	1	1	8	F
0	0	0	8	0	0	0	0	0	0	0	0	1	0	0	0	0	1	1	0	0	0	0	1	6	1
0	4	0	0	0	0	0	0	0	1	0	0	0	0	0	0	x	x	x	x	x	x	x	x	x	x
0	4	0	1	0	0	0	0	0	1	0	0	0	0	0	1	x	x	x	x	x	x	x	x	x	x
0	4	0	2	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	1	1	0	3
0	4	0	3	0	0	0	0	0	1	0	0	0	0	1	1	x	x	x	x	x	x	x	x	x	x

PC	0x0004

R0	0x03
R1	0xXX

R2	0x04
R3	0x02

Z

Fetched data	0x81
Decoded Instruction	

Clk 4

Fetch and increment PC

Program Execution

Processor Z

ADDRESS																DATA									
Hex				Binary												Binary								Hex	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	1	0	9	2
0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	1	0	1	1	8	B
0	0	0	2	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	2	0
0	0	0	3	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	1	8	1
0	0	0	4	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	4
0	0	0	5	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0	0	0	0	0	0	4	0
0	0	0	6	0	0	0	0	0	0	0	0	0	1	1	0	1	1	1	1	0	1	0	0	F	4
0	0	0	7	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	1	1	1	1	8	F
0	0	0	8	0	0	0	0	0	0	0	0	1	0	0	0	0	1	1	0	0	0	0	1	6	1
0	4	0	0	0	0	0	0	0	1	0	0	0	0	0	0	x	x	x	x	x	x	x	x	x	x
0	4	0	1	0	0	0	0	0	1	0	0	0	0	0	1	x	x	x	x	x	x	x	x	x	x
0	4	0	2	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	1	1	0	3
0	4	0	3	0	0	0	0	0	1	0	0	0	0	1	1	x	x	x	x	x	x	x	x	x	x

PC	0x0004

R0	0x03
R1	0xXX

R2	0x04
R3	0x02

Z

Fetched data	0x81
Decoded Instruction	
10 0000 01	
Ldi R1, 0	

Clk 4
Decode

Assembly instruction mnemonic		Machine Code (binary)	Action
ldz	reg	001 000 rr	Load register from memory location pointed to by register Z
ldi	reg, vv	10 vvvv rr	Load immediate register with value v
stz	reg	011 000 rr	Store register to memory location pointed to by register Z
add	regd, regs	000 0 rd rs	Add regd to regs and place result in regd (regd ← regd + regs)
dec	reg	010 000 rr	Decrement the contents of register
bnz	reg, offset	11 vvvv rr	If the value in reg is not zero, branch to the location given by PC + offset (vvvv)

Program Execution

Processor Z

ADDRESS																DATA									
Hex				Binary												Binary								Hex	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	1	0	9	2
0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	1	0	1	1	8	B
0	0	0	2	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	2	0
0	0	0	3	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	1	8	1
0	0	0	4	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	4
0	0	0	5	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0	0	0	0	0	0	4	0
0	0	0	6	0	0	0	0	0	0	0	0	0	1	1	0	1	1	1	1	0	1	0	0	F	4
0	0	0	7	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	1	1	1	1	8	F
0	0	0	8	0	0	0	0	0	0	0	0	1	0	0	0	0	1	1	0	0	0	0	1	6	1
0	4	0	0	0	0	0	0	0	1	0	0	0	0	0	0	x	x	x	x	x	x	x	x	x	x
0	4	0	1	0	0	0	0	0	1	0	0	0	0	0	1	x	x	x	x	x	x	x	x	x	x
0	4	0	2	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	1	1	0	3
0	4	0	3	0	0	0	0	0	1	0	0	0	0	1	1	x	x	x	x	x	x	x	x	x	x

PC	0x0004

R0	0x03	R2	0x04
R1	0x00	R3	0x02

Z

Fetched data	0x81
Decoded Instruction	
10 0000 01	
Ldi R1, 0	

Clk 4
Execute

Program Execution

Processor Z

ADDRESS																DATA									
Hex				Binary												Binary								Hex	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	1	0	9	2
0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	1	0	1	1	8	B
0	0	0	2	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	2	0
0	0	0	3	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	1	8	1
0	0	0	4	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	4
0	0	0	5	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0	0	0	0	0	0	4	0
0	0	0	6	0	0	0	0	0	0	0	0	0	1	1	0	1	1	1	1	0	1	0	0	F	4
0	0	0	7	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	1	1	1	1	8	F
0	0	0	8	0	0	0	0	0	0	0	0	1	0	0	0	0	1	1	0	0	0	0	1	6	1
0	4	0	0	0	0	0	0	0	1	0	0	0	0	0	0	x	x	x	x	x	x	x	x	x	x
0	4	0	1	0	0	0	0	0	1	0	0	0	0	0	1	x	x	x	x	x	x	x	x	x	x
0	4	0	2	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	1	1	0	3
0	4	0	3	0	0	0	0	0	1	0	0	0	0	1	1	x	x	x	x	x	x	x	x	x	x

PC	0x0005

R0	0x03
R1	0x00

R2	0x04
R3	0x02

Z

Fetched data	0x04
Decoded Instruction	

Clk 5

Fetch and increment PC

Program Execution

Processor Z

ADDRESS																DATA									
Hex				Binary												Binary								Hex	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	1	0	9	2
0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	1	0	1	1	8	B
0	0	0	2	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	2	0
0	0	0	3	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	1	8	1
0	0	0	4	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	4
0	0	0	5	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0	0	0	0	0	0	4	0
0	0	0	6	0	0	0	0	0	0	0	0	0	1	1	0	1	1	1	1	0	1	0	0	F	4
0	0	0	7	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	1	1	1	1	8	F
0	0	0	8	0	0	0	0	0	0	0	0	1	0	0	0	0	1	1	0	0	0	0	1	6	1
0	4	0	0	0	0	0	0	0	1	0	0	0	0	0	0	x	x	x	x	x	x	x	x	x	x
0	4	0	1	0	0	0	0	0	1	0	0	0	0	0	1	x	x	x	x	x	x	x	x	x	x
0	4	0	2	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	1	1	0	3
0	4	0	3	0	0	0	0	0	1	0	0	0	0	1	1	x	x	x	x	x	x	x	x	x	x

PC	0x0005

R0	0x03
R1	0x00

R2	0x04
R3	0x02

Z

Fetched data	0x04
Decoded Instruction	
000 0 01 00	
Add R1, R0	

Clk 5
Decode

Assembly instruction mnemonic		Machine Code (binary)	Action
ldz	reg	001 000 rr	Load register from memory location pointed to by register Z
ldi	reg, vv	10 vvvv rr	Load immediate register with value v
stz	reg	011 000 rr	Store register to memory location pointed to by register Z
add	regd, regs	000 0 rd rs	Add regd to regs and place result in regd (regd ← regd + regs)
dec	reg	010 000 rr	Decrement the contents of register
bnz	reg, offset	11 vvvv rr	If the value in reg is not zero, branch to the location given by PC + offset (vvvv)

Program Execution

Processor Z

ADDRESS																DATA									
Hex				Binary												Binary								Hex	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	1	0	9	2
0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	1	0	1	1	8	B
0	0	0	2	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	2	0
0	0	0	3	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	1	8	1
0	0	0	4	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	4
0	0	0	5	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0	0	0	0	0	0	4	0
0	0	0	6	0	0	0	0	0	0	0	0	0	1	1	0	1	1	1	1	0	1	0	0	F	4
0	0	0	7	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	1	1	1	1	8	F
0	0	0	8	0	0	0	0	0	0	0	0	1	0	0	0	0	1	1	0	0	0	0	1	6	1
0	4	0	0	0	0	0	0	0	1	0	0	0	0	0	0	x	x	x	x	x	x	x	x	x	x
0	4	0	1	0	0	0	0	0	1	0	0	0	0	0	1	x	x	x	x	x	x	x	x	x	x
0	4	0	2	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	1	1	0	3
0	4	0	3	0	0	0	0	0	1	0	0	0	0	1	1	x	x	x	x	x	x	x	x	x	x

PC	0x0005

R0	0x03
R1	0x03

R2	0x04
R3	0x02

Z

Fetched data	0x04
Decoded Instruction	
000 0 01 00	
Add R1, R0	

Clk 5
Execute

Program Execution

Processor Z

ADDRESS																DATA									
Hex				Binary												Binary								Hex	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	1	0	9	2
0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	1	0	1	1	8	B
0	0	0	2	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	2	0
0	0	0	3	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	1	8	1
0	0	0	4	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	4
0	0	0	5	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0	0	0	0	0	0	4	0
0	0	0	6	0	0	0	0	0	0	0	0	0	1	1	0	1	1	1	1	0	1	0	0	F	4
0	0	0	7	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	1	1	1	1	8	F
0	0	0	8	0	0	0	0	0	0	0	0	1	0	0	0	0	1	1	0	0	0	0	1	6	1
0	4	0	0	0	0	0	0	0	1	0	0	0	0	0	0	x	x	x	x	x	x	x	x	x	x
0	4	0	1	0	0	0	0	0	1	0	0	0	0	0	1	x	x	x	x	x	x	x	x	x	x
0	4	0	2	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	1	1	0	3
0	4	0	3	0	0	0	0	0	1	0	0	0	0	1	1	x	x	x	x	x	x	x	x	x	x

PC	0x0006

R0	0x03
R1	0x03

R2	0x04
R3	0x02

Z

Fetched data	0x40
Decoded Instruction	

Clk 6

Fetch and increment PC

Program Execution

Processor Z

ADDRESS																	DATA									
Hex				Binary													Binary								Hex	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	1	0	9	2
0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	1	0	1	1	8	B
0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	2	0
0	0	0	3	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	1	8	1
0	0	0	4	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	4
0	0	0	5	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0	0	0	0	0	4	0
0	0	0	6	0	0	0	0	0	0	0	0	0	0	0	1	1	0	1	1	1	1	0	1	0	0	F	4
0	0	0	7	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	1	1	1	1	8	F
0	0	0	8	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	1	0	0	0	0	1	6	1
0	4	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	x	x	x	x	x	x	x	x	x	x
0	4	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	1	x	x	x	x	x	x	x	x	x	x
0	4	0	2	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	1	0	3
0	4	0	3	0	0	0	0	0	1	0	0	0	0	0	0	1	1	x	x	x	x	x	x	x	x	x	x

PC	0x0006

R0	0x03	R2	0x04
R1	0x03	R3	0x02

Z

Fetched data	0x40
Decoded Instruction	
101 000 00	
Dec R0	

Clk 6
Decode

Assembly instruction mnemonic		Machine Code (binary)	Action
ldz	reg	001 000 rr	Load register from memory location pointed to by register Z
ldi	reg, vv	10 vvvv rr	Load immediate register with value v
stz	reg	011 000 rr	Store register to memory location pointed to by register Z
add	regd, regs	000 0 rd rs	Add regd to regs and place result in regd (regd ← regd + regs)
dec	reg	010 000 rr	Decrement the contents of register
bnz	reg, offset	11 vvvv rr	If the value in reg is not zero, branch to the location given by PC + offset (vvvv)

Program Execution

Processor Z

ADDRESS																DATA									
Hex				Binary												Binary								Hex	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	1	0	9	2
0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	1	0	1	1	8	B
0	0	0	2	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	2	0
0	0	0	3	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	1	8	1
0	0	0	4	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	4
0	0	0	5	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0	0	0	0	0	0	4	0
0	0	0	6	0	0	0	0	0	0	0	0	0	1	1	0	1	1	1	1	0	1	0	0	F	4
0	0	0	7	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	1	1	1	1	8	F
0	0	0	8	0	0	0	0	0	0	0	0	1	0	0	0	0	1	1	0	0	0	0	1	6	1
0	4	0	0	0	0	0	0	0	1	0	0	0	0	0	0	x	x	x	x	x	x	x	x	x	x
0	4	0	1	0	0	0	0	0	1	0	0	0	0	0	1	x	x	x	x	x	x	x	x	x	x
0	4	0	2	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	1	1	0	3
0	4	0	3	0	0	0	0	0	1	0	0	0	0	1	1	x	x	x	x	x	x	x	x	x	x

PC	0x0006

R0	0x02
R1	0x03

R2	0x04
R3	0x02

Z

Fetched data	0x40
Decoded Instruction	
101 000 00	
Dec R0	

Clk 6
Execute

Program Execution

Processor Z

ADDRESS																DATA									
Hex				Binary												Binary								Hex	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	1	0	9	2
0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	1	0	1	1	8	B
0	0	0	2	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	2	0
0	0	0	3	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	1	8	1
0	0	0	4	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	4
0	0	0	5	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0	0	0	0	0	0	4	0
0	0	0	6	0	0	0	0	0	0	0	0	0	1	1	0	1	1	1	1	0	1	0	0	F	4
0	0	0	7	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	1	1	1	1	8	F
0	0	0	8	0	0	0	0	0	0	0	0	1	0	0	0	0	1	1	0	0	0	0	1	6	1
0	4	0	0	0	0	0	0	0	1	0	0	0	0	0	0	x	x	x	x	x	x	x	x	x	x
0	4	0	1	0	0	0	0	0	1	0	0	0	0	0	1	x	x	x	x	x	x	x	x	x	x
0	4	0	2	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	1	1	0	3
0	4	0	3	0	0	0	0	0	1	0	0	0	0	1	1	x	x	x	x	x	x	x	x	x	x

PC	0x0007

R0	0x02
R1	0x03

R2	0x04
R3	0x02

Z

Fetched data	0xF4
Decoded Instruction	

Clk 7

Fetch and increment PC

Program Execution

Processor Z

ADDRESS																DATA									
Hex				Binary												Binary								Hex	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	1	0	9	2
0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	1	0	1	1	8	B
0	0	0	2	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	2	0
0	0	0	3	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	1	8	1
0	0	0	4	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	4
0	0	0	5	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0	0	0	0	0	0	4	0
0	0	0	6	0	0	0	0	0	0	0	0	0	1	1	0	1	1	1	1	0	1	0	0	F	4
0	0	0	7	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	1	1	1	1	8	F
0	0	0	8	0	0	0	0	0	0	0	0	1	0	0	0	0	1	1	0	0	0	0	1	6	1
0	4	0	0	0	0	0	0	0	1	0	0	0	0	0	0	x	x	x	x	x	x	x	x	x	x
0	4	0	1	0	0	0	0	0	1	0	0	0	0	0	1	x	x	x	x	x	x	x	x	x	x
0	4	0	2	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	1	1	0	3
0	4	0	3	0	0	0	0	0	1	0	0	0	0	1	1	x	x	x	x	x	x	x	x	x	x

PC	0x0007

R0	0x02
R1	0x03

R2	0x04
R3	0x02

Z

Fetched data	0xF4
Decoded Instruction	
11 1101 00	
Bnz R0, -3	

Note: 1101 2's comp = -3

Clk 7
Decode

Assembly instruction mnemonic		Machine Code (binary)	Action
ldz	reg	001 000 rr	Load register from memory location pointed to by register Z
ldi	reg, vv	10 vvvv rr	Load immediate register with value v
stz	reg	011 000 rr	Store register to memory location pointed to by register Z
add	regd, regs	000 0 rd rs	Add regd to regs and place result in regd (regd ← regd + regs)
dec	reg	010 000 rr	Decrement the contents of register
bnz	reg, offset	11 vvvv rr	If the value in reg is not zero, branch to the location given by PC + offset (vvvv)

Program Execution

Processor Z

ADDRESS																DATA									
Hex				Binary												Binary								Hex	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	1	0	9	2
0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	1	0	1	1	8	B
0	0	0	2	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	2	0
0	0	0	3	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	1	8	1
0	0	0	4	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	4
0	0	0	5	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0	0	0	0	0	0	4	0
0	0	0	6	0	0	0	0	0	0	0	0	0	1	1	0	1	1	1	1	0	1	0	0	F	4
0	0	0	7	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	1	1	1	1	8	F
0	0	0	8	0	0	0	0	0	0	0	0	1	0	0	0	0	1	1	0	0	0	0	1	6	1
0	4	0	0	0	0	0	0	0	1	0	0	0	0	0	0	x	x	x	x	x	x	x	x	x	x
0	4	0	1	0	0	0	0	0	1	0	0	0	0	0	1	x	x	x	x	x	x	x	x	x	x
0	4	0	2	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	1	1	0	3
0	4	0	3	0	0	0	0	0	1	0	0	0	0	1	1	x	x	x	x	x	x	x	x	x	x

PC	0x0004

R0	0x02
R1	0x03

R2	0x04
R3	0x02

Z

Fetched data	0xF4
Decoded Instruction	
11 1101 00	
Bnz R0, -3	

Note: PC reset to 0x0004

Clk 7
Execute

Program Execution

Processor Z

ADDRESS																DATA									
Hex				Binary												Binary								Hex	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	1	0	9	2
0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	1	0	1	1	8	B
0	0	0	2	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	2	0
0	0	0	3	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	1	8	1
0	0	0	4	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	4
0	0	0	5	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0	0	0	0	0	0	4	0
0	0	0	6	0	0	0	0	0	0	0	0	0	1	1	0	1	1	1	1	0	1	0	0	F	4
0	0	0	7	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	1	1	1	1	8	F
0	0	0	8	0	0	0	0	0	0	0	0	1	0	0	0	0	1	1	0	0	0	0	1	6	1
0	4	0	0	0	0	0	0	0	1	0	0	0	0	0	0	x	x	x	x	x	x	x	x	x	x
0	4	0	1	0	0	0	0	0	1	0	0	0	0	0	1	x	x	x	x	x	x	x	x	x	x
0	4	0	2	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	1	1	0	3
0	4	0	3	0	0	0	0	0	1	0	0	0	0	1	1	x	x	x	x	x	x	x	x	x	x

PC	0x0005

R0	0x02
R1	0x03

R2	0x04
R3	0x02

Z

Fetched data	0x04
Decoded Instruction	

Note: PC was pointing to 0x0004

Clk 8

Fetch and increment PC

Program Execution

Processor Z

ADDRESS																DATA									
Hex				Binary												Binary								Hex	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	1	0	9	2
0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	1	0	1	1	8	B
0	0	0	2	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	2	0
0	0	0	3	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	1	8	1
0	0	0	4	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	4
0	0	0	5	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0	0	0	0	0	0	4	0
0	0	0	6	0	0	0	0	0	0	0	0	0	1	1	0	1	1	1	1	0	1	0	0	F	4
0	0	0	7	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	1	1	1	1	8	F
0	0	0	8	0	0	0	0	0	0	0	0	1	0	0	0	0	1	1	0	0	0	0	1	6	1
0	4	0	0	0	0	0	0	0	1	0	0	0	0	0	0	x	x	x	x	x	x	x	x	x	x
0	4	0	1	0	0	0	0	0	1	0	0	0	0	0	1	x	x	x	x	x	x	x	x	x	x
0	4	0	2	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	1	1	0	3
0	4	0	3	0	0	0	0	0	1	0	0	0	0	1	1	x	x	x	x	x	x	x	x	x	x

PC	0x0005

R0	0x02
R1	0x03

R2	0x04
R3	0x02

Z

Fetched data	0x04
Decoded Instruction	
000 0 01 00	
Add R1, R0	

Clk 8
Decode

Assembly instruction mnemonic		Machine Code (binary)	Action
ldz	reg	001 000 rr	Load register from memory location pointed to by register Z
ldi	reg, vv	10 vvvv rr	Load immediate register with value v
stz	reg	011 000 rr	Store register to memory location pointed to by register Z
add	regd, regs	000 0 rd rs	Add regd to regs and place result in regd (regd ← regd + regs)
dec	reg	010 000 rr	Decrement the contents of register
bnz	reg, offset	11 vvvv rr	If the value in reg is not zero, branch to the location given by PC + offset (vvvv)

Program Execution

Processor Z

ADDRESS																	DATA									
Hex				Binary													Binary								Hex	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	1	0	9	2
0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	1	0	1	1	8	B
0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	2	0
0	0	0	3	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	1	8	1
0	0	0	4	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	4
0	0	0	5	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0	0	0	0	0	4	0
0	0	0	6	0	0	0	0	0	0	0	0	0	0	0	1	1	0	1	1	1	1	0	1	0	F	4
0	0	0	7	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	1	1	1	8	F
0	0	0	8	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	1	0	0	0	1	6	1
0	4	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	x	x	x	x	x	x	x	x	x
0	4	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	1	x	x	x	x	x	x	x	x	x
0	4	0	2	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	1	1	0	3
0	4	0	3	0	0	0	0	0	1	0	0	0	0	0	0	1	1	x	x	x	x	x	x	x	x	x

PC	0x0005

R0	0x02
R1	0x05

R2	0x04
R3	0x02

Z

Fetched data	0x04
Decoded Instruction	
000 0 01 00	
Add R1, R0	

Clk 8
Execute

Program Execution

Processor Z

ADDRESS																DATA									
Hex				Binary												Binary								Hex	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	1	0	9	2
0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	1	0	1	1	8	B
0	0	0	2	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	2	0
0	0	0	3	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	1	8	1
0	0	0	4	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	4
0	0	0	5	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0	0	0	0	0	0	4	0
0	0	0	6	0	0	0	0	0	0	0	0	0	1	1	0	1	1	1	1	0	1	0	0	F	4
0	0	0	7	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	1	1	1	1	8	F
0	0	0	8	0	0	0	0	0	0	0	0	1	0	0	0	0	1	1	0	0	0	0	1	6	1
0	4	0	0	0	0	0	0	0	1	0	0	0	0	0	0	x	x	x	x	x	x	x	x	x	x
0	4	0	1	0	0	0	0	0	1	0	0	0	0	0	1	x	x	x	x	x	x	x	x	x	x
0	4	0	2	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	1	1	0	3
0	4	0	3	0	0	0	0	0	1	0	0	0	0	1	1	x	x	x	x	x	x	x	x	x	x

PC	0x0006

R0	0x02
R1	0x05

R2	0x04
R3	0x02

Z

Fetched data	0x40
Decoded Instruction	

Clk 9

Fetch and increment PC

Program Execution

Processor Z

ADDRESS																DATA									
Hex				Binary												Binary								Hex	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	1	0	9	2
0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	1	0	1	1	8	B
0	0	0	2	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	2	0
0	0	0	3	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	1	8	1
0	0	0	4	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	4
0	0	0	5	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0	0	0	0	0	0	4	0
0	0	0	6	0	0	0	0	0	0	0	0	0	1	1	0	1	1	1	1	0	1	0	0	F	4
0	0	0	7	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	1	1	1	1	8	F
0	0	0	8	0	0	0	0	0	0	0	0	1	0	0	0	0	1	1	0	0	0	0	1	6	1
0	4	0	0	0	0	0	0	0	1	0	0	0	0	0	0	x	x	x	x	x	x	x	x	x	x
0	4	0	1	0	0	0	0	0	1	0	0	0	0	0	1	x	x	x	x	x	x	x	x	x	x
0	4	0	2	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	1	1	0	3
0	4	0	3	0	0	0	0	0	1	0	0	0	0	1	1	x	x	x	x	x	x	x	x	x	x

PC	0x0006

R0	0x02
R1	0x05

R2	0x04
R3	0x02

Z

Fetched data	0x40
Decoded Instruction	
101 000 00	
Dec R0	

Clk 9
Decode

Assembly instruction mnemonic		Machine Code (binary)	Action
ldz	reg	001 000 rr	Load register from memory location pointed to by register Z
ldi	reg, vv	10 vvvv rr	Load immediate register with value v
stz	reg	011 000 rr	Store register to memory location pointed to by register Z
add	regd, regs	000 0 rd rs	Add regd to regs and place result in regd (regd ← regd + regs)
dec	reg	010 000 rr	Decrement the contents of register
bnz	reg, offset	11 vvvv rr	If the value in reg is not zero, branch to the location given by PC + offset (vvvv)

Program Execution

Processor Z

ADDRESS																DATA									
Hex				Binary												Binary								Hex	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	1	0	9	2
0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	1	0	1	1	8	B
0	0	0	2	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	2	0
0	0	0	3	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	1	8	1
0	0	0	4	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	4
0	0	0	5	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0	0	0	0	0	0	4	0
0	0	0	6	0	0	0	0	0	0	0	0	0	1	1	0	1	1	1	1	0	1	0	0	F	4
0	0	0	7	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	1	1	1	1	8	F
0	0	0	8	0	0	0	0	0	0	0	0	1	0	0	0	0	1	1	0	0	0	0	1	6	1
0	4	0	0	0	0	0	0	0	1	0	0	0	0	0	0	x	x	x	x	x	x	x	x	x	x
0	4	0	1	0	0	0	0	0	1	0	0	0	0	0	1	x	x	x	x	x	x	x	x	x	x
0	4	0	2	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	1	1	0	3
0	4	0	3	0	0	0	0	0	1	0	0	0	0	1	1	x	x	x	x	x	x	x	x	x	x

PC	0x0006

R0	0x01
R1	0x05

R2	0x04
R3	0x02

Z

Fetched data	0x40
Decoded Instruction	
101 000 00	
Dec R0	

Clk 9
Execute

Program Execution

Processor Z

ADDRESS																DATA									
Hex				Binary												Binary								Hex	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	1	0	9	2
0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	1	0	1	1	8	B
0	0	0	2	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	2	0
0	0	0	3	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	1	8	1
0	0	0	4	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	4
0	0	0	5	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0	0	0	0	0	0	4	0
0	0	0	6	0	0	0	0	0	0	0	0	0	1	1	0	1	1	1	1	0	1	0	0	F	4
0	0	0	7	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	1	1	1	1	8	F
0	0	0	8	0	0	0	0	0	0	0	0	1	0	0	0	0	1	1	0	0	0	0	1	6	1
0	4	0	0	0	0	0	0	0	1	0	0	0	0	0	0	x	x	x	x	x	x	x	x	x	x
0	4	0	1	0	0	0	0	0	1	0	0	0	0	0	1	x	x	x	x	x	x	x	x	x	x
0	4	0	2	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	1	1	0	3
0	4	0	3	0	0	0	0	0	1	0	0	0	0	1	1	x	x	x	x	x	x	x	x	x	x

PC	0x0007

R0	0x01
R1	0x05

R2	0x04
R3	0x02

Z

Fetched data	0xF4
Decoded Instruction	

Clk 10

Fetch and increment PC

Program Execution

Processor Z

ADDRESS																DATA									
Hex				Binary												Binary								Hex	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	1	0	9	2
0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	1	0	1	1	8	B
0	0	0	2	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	2	0
0	0	0	3	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	1	8	1
0	0	0	4	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	4
0	0	0	5	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0	0	0	0	0	0	4	0
0	0	0	6	0	0	0	0	0	0	0	0	0	1	1	0	1	1	1	1	0	1	0	0	F	4
0	0	0	7	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	1	1	1	1	8	F
0	0	0	8	0	0	0	0	0	0	0	0	1	0	0	0	0	1	1	0	0	0	0	1	6	1
0	4	0	0	0	0	0	0	0	1	0	0	0	0	0	0	x	x	x	x	x	x	x	x	x	x
0	4	0	1	0	0	0	0	0	1	0	0	0	0	0	1	x	x	x	x	x	x	x	x	x	x
0	4	0	2	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	1	1	0	3
0	4	0	3	0	0	0	0	0	1	0	0	0	0	1	1	x	x	x	x	x	x	x	x	x	x

PC	0x0007

R0	0x01
R1	0x05

R2	0x04
R3	0x02

Z

Fetched data	0xF4
Decoded Instruction	
11 1101 00	
Bnz R0, -3	

Clk 10
Decode

Assembly instruction mnemonic		Machine Code (binary)	Action
ldz	reg	001 000 rr	Load register from memory location pointed to by register Z
ldi	reg, vv	10 vvvv rr	Load immediate register with value v
stz	reg	011 000 rr	Store register to memory location pointed to by register Z
add	regd, regs	000 0 rd rs	Add regd to regs and place result in regd (regd ← regd + regs)
dec	reg	010 000 rr	Decrement the contents of register
bnz	reg, offset	11 vvvv rr	If the value in reg is not zero, branch to the location given by PC + offset (vvvv)

Program Execution

Processor Z

ADDRESS																DATA									
Hex				Binary												Binary								Hex	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	1	0	9	2
0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	1	0	1	1	8	B
0	0	0	2	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	2	0
0	0	0	3	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	1	8	1
0	0	0	4	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	4
0	0	0	5	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0	0	0	0	0	0	4	0
0	0	0	6	0	0	0	0	0	0	0	0	0	1	1	0	1	1	1	1	0	1	0	0	F	4
0	0	0	7	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	1	1	1	1	8	F
0	0	0	8	0	0	0	0	0	0	0	0	1	0	0	0	0	1	1	0	0	0	0	1	6	1
0	4	0	0	0	0	0	0	0	1	0	0	0	0	0	0	x	x	x	x	x	x	x	x	x	x
0	4	0	1	0	0	0	0	0	1	0	0	0	0	0	1	x	x	x	x	x	x	x	x	x	x
0	4	0	2	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	1	1	0	3
0	4	0	3	0	0	0	0	0	1	0	0	0	0	1	1	x	x	x	x	x	x	x	x	x	x

PC	0x0004

R0	0x01
R1	0x05

R2	0x04
R3	0x02

Z

Fetched data	0xF4
Decoded Instruction	
11 1101 00	
Bnz R0, -3	

Note: PC was reset to 0x0004

Clk 10
Execute

Program Execution

Processor Z

ADDRESS																DATA									
Hex				Binary												Binary								Hex	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	1	0	9	2
0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	1	0	1	1	8	B
0	0	0	2	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	2	0
0	0	0	3	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	1	8	1
0	0	0	4	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	4
0	0	0	5	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0	0	0	0	0	0	4	0
0	0	0	6	0	0	0	0	0	0	0	0	0	1	1	0	1	1	1	1	0	1	0	0	F	4
0	0	0	7	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	1	1	1	1	8	F
0	0	0	8	0	0	0	0	0	0	0	0	1	0	0	0	0	1	1	0	0	0	0	1	6	1
0	4	0	0	0	0	0	0	0	1	0	0	0	0	0	0	x	x	x	x	x	x	x	x	x	x
0	4	0	1	0	0	0	0	0	1	0	0	0	0	0	1	x	x	x	x	x	x	x	x	x	x
0	4	0	2	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	1	1	0	3
0	4	0	3	0	0	0	0	0	1	0	0	0	0	1	1	x	x	x	x	x	x	x	x	x	x

PC	0x0005

R0	0x01
R1	0x05

R2	0x04
R3	0x02

Z

Fetched data	0x04
Decoded Instruction	

Note: PC was pointing to 0x0004

Clk 11
Fetch and increment PC

Program Execution

Processor Z

ADDRESS																DATA									
Hex				Binary												Binary								Hex	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	1	0	9	2
0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	1	0	1	1	8	B
0	0	0	2	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	2	0
0	0	0	3	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	1	8	1
0	0	0	4	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	4
0	0	0	5	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0	0	0	0	0	0	4	0
0	0	0	6	0	0	0	0	0	0	0	0	0	1	1	0	1	1	1	1	0	1	0	0	F	4
0	0	0	7	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	1	1	1	1	8	F
0	0	0	8	0	0	0	0	0	0	0	0	1	0	0	0	0	1	1	0	0	0	0	1	6	1
0	4	0	0	0	0	0	0	0	1	0	0	0	0	0	0	x	x	x	x	x	x	x	x	x	x
0	4	0	1	0	0	0	0	0	1	0	0	0	0	0	1	x	x	x	x	x	x	x	x	x	x
0	4	0	2	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	1	1	0	3
0	4	0	3	0	0	0	0	0	1	0	0	0	0	1	1	x	x	x	x	x	x	x	x	x	x

PC	0x0005

R0	0x01
R1	0x05

R2	0x04
R3	0x02

Z

Fetched data	0x04
Decoded Instruction	
000 0 01 00	
Add R1, R0	

Clk 11
Decode

Assembly instruction mnemonic		Machine Code (binary)	Action
ldz	reg	001 000 rr	Load register from memory location pointed to by register Z
ldi	reg, vv	10 vvvv rr	Load immediate register with value v
stz	reg	011 000 rr	Store register to memory location pointed to by register Z
add	regd, regs	000 0 rd rs	Add regd to regs and place result in regd (regd ← regd + regs)
dec	reg	010 000 rr	Decrement the contents of register
bnz	reg, offset	11 vvvv rr	If the value in reg is not zero, branch to the location given by PC + offset (vvvv)

Program Execution

Processor Z

ADDRESS																DATA									
Hex				Binary												Binary								Hex	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	1	0	9	2
0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	1	0	1	1	8	B
0	0	0	2	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	2	0
0	0	0	3	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	1	8	1
0	0	0	4	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	4
0	0	0	5	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0	0	0	0	0	0	4	0
0	0	0	6	0	0	0	0	0	0	0	0	0	1	1	0	1	1	1	1	0	1	0	0	F	4
0	0	0	7	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	1	1	1	1	8	F
0	0	0	8	0	0	0	0	0	0	0	0	1	0	0	0	0	1	1	0	0	0	0	1	6	1
0	4	0	0	0	0	0	0	0	1	0	0	0	0	0	0	x	x	x	x	x	x	x	x	x	x
0	4	0	1	0	0	0	0	0	1	0	0	0	0	0	1	x	x	x	x	x	x	x	x	x	x
0	4	0	2	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	1	1	0	3
0	4	0	3	0	0	0	0	0	1	0	0	0	0	1	1	x	x	x	x	x	x	x	x	x	x

PC	0x0005

R0	0x01
R1	0x06

R2	0x04
R3	0x02

Z

Fetched data	0x04
Decoded Instruction	
000 0 01 00	
Add R1, R0	

Clk 11
Execute

Program Execution

Processor Z

ADDRESS																DATA									
Hex				Binary												Binary								Hex	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	1	0	9	2
0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	1	0	1	1	8	B
0	0	0	2	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	2	0
0	0	0	3	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	1	8	1
0	0	0	4	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	4
0	0	0	5	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0	0	0	0	0	0	4	0
0	0	0	6	0	0	0	0	0	0	0	0	0	1	1	0	1	1	1	1	0	1	0	0	F	4
0	0	0	7	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	1	1	1	1	8	F
0	0	0	8	0	0	0	0	0	0	0	0	1	0	0	0	0	1	1	0	0	0	0	1	6	1
0	4	0	0	0	0	0	0	0	1	0	0	0	0	0	0	x	x	x	x	x	x	x	x	x	x
0	4	0	1	0	0	0	0	0	1	0	0	0	0	0	1	x	x	x	x	x	x	x	x	x	x
0	4	0	2	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	1	1	0	3
0	4	0	3	0	0	0	0	0	1	0	0	0	0	1	1	x	x	x	x	x	x	x	x	x	x

PC	0x0006

R0	0x01
R1	0x06

R2	0x04
R3	0x02

Z

Fetched data	0x40
Decoded Instruction	

Clk 12

Fetch and increment PC

Program Execution

Processor Z

ADDRESS																DATA									
Hex				Binary												Binary								Hex	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	1	0	9	2
0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	1	0	1	1	8	B
0	0	0	2	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	2	0
0	0	0	3	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	1	8	1
0	0	0	4	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	4
0	0	0	5	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0	0	0	0	0	0	4	0
0	0	0	6	0	0	0	0	0	0	0	0	0	1	1	0	1	1	1	1	0	1	0	0	F	4
0	0	0	7	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	1	1	1	1	8	F
0	0	0	8	0	0	0	0	0	0	0	0	1	0	0	0	0	1	1	0	0	0	0	1	6	1
0	4	0	0	0	0	0	0	0	1	0	0	0	0	0	0	x	x	x	x	x	x	x	x	x	x
0	4	0	1	0	0	0	0	0	1	0	0	0	0	0	1	x	x	x	x	x	x	x	x	x	x
0	4	0	2	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	1	1	0	3
0	4	0	3	0	0	0	0	0	1	0	0	0	0	1	1	x	x	x	x	x	x	x	x	x	x

PC	0x0006

R0	0x01
R1	0x06

R2	0x04
R3	0x02

Z

Fetched data	0x40
Decoded Instruction	
101 000 00	
Dec R0	

Clk 12
Decode

Assembly instruction mnemonic		Machine Code (binary)	Action
ldz	reg	001 000 rr	Load register from memory location pointed to by register Z
ldi	reg, vv	10 vvvv rr	Load immediate register with value v
stz	reg	011 000 rr	Store register to memory location pointed to by register Z
add	regd, regs	000 0 rd rs	Add regd to regs and place result in regd (regd ← regd + regs)
dec	reg	010 000 rr	Decrement the contents of register
bnz	reg, offset	11 vvvv rr	If the value in reg is not zero, branch to the location given by PC + offset (vvvv)

Program Execution

Processor Z

ADDRESS																DATA									
Hex				Binary												Binary								Hex	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	1	0	9	2
0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	1	0	1	1	8	B
0	0	0	2	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	2	0
0	0	0	3	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	1	8	1
0	0	0	4	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	4
0	0	0	5	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0	0	0	0	0	0	4	0
0	0	0	6	0	0	0	0	0	0	0	0	0	1	1	0	1	1	1	1	0	1	0	0	F	4
0	0	0	7	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	1	1	1	1	8	F
0	0	0	8	0	0	0	0	0	0	0	0	1	0	0	0	0	1	1	0	0	0	0	1	6	1
0	4	0	0	0	0	0	0	0	1	0	0	0	0	0	0	x	x	x	x	x	x	x	x	x	x
0	4	0	1	0	0	0	0	0	1	0	0	0	0	0	1	x	x	x	x	x	x	x	x	x	x
0	4	0	2	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	1	1	0	3
0	4	0	3	0	0	0	0	0	1	0	0	0	0	1	1	x	x	x	x	x	x	x	x	x	x

PC	0x0006

R0	0x00
R1	0x06

R2	0x04
R3	0x02

Z

Fetched data	0x40
Decoded Instruction	
101 000 00	
Dec R0	

Clk 12
Execute

Program Execution

Processor Z

ADDRESS																DATA									
Hex				Binary												Binary								Hex	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	1	0	9	2
0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	1	0	1	1	8	B
0	0	0	2	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	2	0
0	0	0	3	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	1	8	1
0	0	0	4	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	4
0	0	0	5	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0	0	0	0	0	0	4	0
0	0	0	6	0	0	0	0	0	0	0	0	0	1	1	0	1	1	1	1	0	1	0	0	F	4
0	0	0	7	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	1	1	1	1	8	F
0	0	0	8	0	0	0	0	0	0	0	0	1	0	0	0	0	1	1	0	0	0	0	1	6	1
0	4	0	0	0	0	0	0	0	1	0	0	0	0	0	0	x	x	x	x	x	x	x	x	x	x
0	4	0	1	0	0	0	0	0	1	0	0	0	0	0	1	x	x	x	x	x	x	x	x	x	x
0	4	0	2	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	1	1	0	3
0	4	0	3	0	0	0	0	0	1	0	0	0	0	1	1	x	x	x	x	x	x	x	x	x	x

PC	0x0007

R0	0x00
R1	0x06

R2	0x04
R3	0x02

Z

Fetched data	0xF4
Decoded Instruction	

Clk 13

Fetch and increment PC

Program Execution

Processor Z

ADDRESS																DATA									
Hex				Binary												Binary								Hex	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	1	0	9	2
0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	1	0	1	1	8	B
0	0	0	2	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	2	0
0	0	0	3	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	1	8	1
0	0	0	4	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	4
0	0	0	5	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0	0	0	0	0	0	4	0
0	0	0	6	0	0	0	0	0	0	0	0	0	1	1	0	1	1	1	1	0	1	0	0	F	4
0	0	0	7	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	1	1	1	1	8	F
0	0	0	8	0	0	0	0	0	0	0	0	1	0	0	0	0	1	1	0	0	0	0	1	6	1
0	4	0	0	0	0	0	0	0	1	0	0	0	0	0	0	x	x	x	x	x	x	x	x	x	x
0	4	0	1	0	0	0	0	0	1	0	0	0	0	0	1	x	x	x	x	x	x	x	x	x	x
0	4	0	2	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	1	1	0	3
0	4	0	3	0	0	0	0	0	1	0	0	0	0	1	1	x	x	x	x	x	x	x	x	x	x

PC	0x0007

R0	0x00
R1	0x06

R2	0x04
R3	0x02

Z

Fetched data	0xF4
Decoded Instruction	
11 1101 00	
Bnz R0, -3	

Clk 13
Decode

Assembly instruction mnemonic		Machine Code (binary)	Action
ldz	reg	001 000 rr	Load register from memory location pointed to by register Z
ldi	reg, vv	10 vvvv rr	Load immediate register with value v
stz	reg	011 000 rr	Store register to memory location pointed to by register Z
add	regd, regs	000 0 rd rs	Add regd to regs and place result in regd (regd ← regd + regs)
dec	reg	010 000 rr	Decrement the contents of register
bnz	reg, offset	11 vvvv rr	If the value in reg is not zero, branch to the location given by PC + offset (vvvv)

Program Execution

Processor Z

ADDRESS																DATA									
Hex				Binary												Binary								Hex	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	1	0	9	2
0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	1	0	1	1	8	B
0	0	0	2	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	2	0
0	0	0	3	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	1	8	1
0	0	0	4	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	4
0	0	0	5	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0	0	0	0	0	0	4	0
0	0	0	6	0	0	0	0	0	0	0	0	0	1	1	0	1	1	1	1	0	1	0	0	F	4
0	0	0	7	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	1	1	1	1	8	F
0	0	0	8	0	0	0	0	0	0	0	0	1	0	0	0	0	1	1	0	0	0	0	1	6	1
0	4	0	0	0	0	0	0	0	1	0	0	0	0	0	0	x	x	x	x	x	x	x	x	x	x
0	4	0	1	0	0	0	0	0	1	0	0	0	0	0	1	x	x	x	x	x	x	x	x	x	x
0	4	0	2	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	1	1	0	3
0	4	0	3	0	0	0	0	0	1	0	0	0	0	1	1	x	x	x	x	x	x	x	x	x	x

PC	0x0007

R0	0x00
R1	0x06

R2	0x04
R3	0x02

Z

Fetched data	0xF4
Decoded Instruction	
11 1101 00	
Bnz R0, -3	

Note: Branch not taken – no action

Clk 13
Execute

Program Execution

Processor Z

ADDRESS																DATA									
Hex				Binary												Binary								Hex	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	1	0	9	2
0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	1	0	1	1	8	B
0	0	0	2	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	2	0
0	0	0	3	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	1	8	1
0	0	0	4	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	4
0	0	0	5	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0	0	0	0	0	0	4	0
0	0	0	6	0	0	0	0	0	0	0	0	0	1	1	0	1	1	1	1	0	1	0	0	F	4
0	0	0	7	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	1	1	1	1	8	F
0	0	0	8	0	0	0	0	0	0	0	0	1	0	0	0	0	1	1	0	0	0	0	1	6	1
0	4	0	0	0	0	0	0	0	1	0	0	0	0	0	0	x	x	x	x	x	x	x	x	x	x
0	4	0	1	0	0	0	0	0	1	0	0	0	0	0	1	x	x	x	x	x	x	x	x	x	x
0	4	0	2	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	1	1	0	3
0	4	0	3	0	0	0	0	0	1	0	0	0	0	1	1	x	x	x	x	x	x	x	x	x	x

PC	0x0008

R0	0x00
R1	0x06

R2	0x04
R3	0x02

Z

Fetched data	0x8F
Decoded Instruction	

Clk 14

Fetch and increment PC

Program Execution

Processor Z

ADDRESS																DATA									
Hex				Binary												Binary								Hex	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	1	0	9	2
0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	1	0	1	1	8	B
0	0	0	2	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	2	0
0	0	0	3	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	1	8	1
0	0	0	4	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	4
0	0	0	5	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0	0	0	0	0	0	4	0
0	0	0	6	0	0	0	0	0	0	0	0	0	1	1	0	1	1	1	1	0	1	0	0	F	4
0	0	0	7	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	1	1	1	1	8	F
0	0	0	8	0	0	0	0	0	0	0	0	1	0	0	0	0	1	1	0	0	0	0	1	6	1
0	4	0	0	0	0	0	0	0	1	0	0	0	0	0	0	x	x	x	x	x	x	x	x	x	x
0	4	0	1	0	0	0	0	0	1	0	0	0	0	0	1	x	x	x	x	x	x	x	x	x	x
0	4	0	2	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	1	1	0	3
0	4	0	3	0	0	0	0	0	1	0	0	0	0	1	1	x	x	x	x	x	x	x	x	x	x

PC	0x0008

R0	0x00
R1	0x06

R2	0x04
R3	0x02

Z

Fetched data	0x8F
Decoded Instruction	
10 0011 11	
Ldi R3,3	

Clk 14
Decode

Assembly instruction mnemonic		Machine Code (binary)	Action
ldz	reg	001 000 rr	Load register from memory location pointed to by register Z
ldi	reg, vv	10 vvvv rr	Load immediate register with value v
stz	reg	011 000 rr	Store register to memory location pointed to by register Z
add	regd, regs	000 0 rd rs	Add regd to regs and place result in regd (regd ← regd + regs)
dec	reg	010 000 rr	Decrement the contents of register
bnz	reg, offset	11 vvvv rr	If the value in reg is not zero, branch to the location given by PC + offset (vvvv)

Program Execution

Processor Z

ADDRESS																DATA									
Hex				Binary												Binary								Hex	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	1	0	9	2
0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	1	0	1	1	8	B
0	0	0	2	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	2	0
0	0	0	3	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	1	8	1
0	0	0	4	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	4
0	0	0	5	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0	0	0	0	0	0	4	0
0	0	0	6	0	0	0	0	0	0	0	0	0	1	1	0	1	1	1	1	0	1	0	0	F	4
0	0	0	7	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	1	1	1	1	8	F
0	0	0	8	0	0	0	0	0	0	0	0	1	0	0	0	0	1	1	0	0	0	0	1	6	1
0	4	0	0	0	0	0	0	0	1	0	0	0	0	0	0	x	x	x	x	x	x	x	x	x	x
0	4	0	1	0	0	0	0	0	1	0	0	0	0	0	1	x	x	x	x	x	x	x	x	x	x
0	4	0	2	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	1	1	0	3
0	4	0	3	0	0	0	0	0	1	0	0	0	0	1	1	x	x	x	x	x	x	x	x	x	x

PC	0x0008

R0	0x00
R1	0x06

R2	0x04
R3	0x03

Z

Fetched data	0x8F
Decoded Instruction	
10 0011 11	
Ldi R3,3	

Clk 14
Execute

Program Execution

Processor Z

ADDRESS																DATA									
Hex				Binary												Binary								Hex	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	1	0	9	2
0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	1	0	1	1	8	B
0	0	0	2	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	2	0
0	0	0	3	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	1	8	1
0	0	0	4	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	4
0	0	0	5	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0	0	0	0	0	0	4	0
0	0	0	6	0	0	0	0	0	0	0	0	0	1	1	0	1	1	1	1	0	1	0	0	F	4
0	0	0	7	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	1	1	1	1	8	F
0	0	0	8	0	0	0	0	0	0	0	0	1	0	0	0	0	1	1	0	0	0	0	1	6	1
0	4	0	0	0	0	0	0	0	1	0	0	0	0	0	0	x	x	x	x	x	x	x	x	x	x
0	4	0	1	0	0	0	0	0	1	0	0	0	0	0	1	x	x	x	x	x	x	x	x	x	x
0	4	0	2	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	1	1	0	3
0	4	0	3	0	0	0	0	0	1	0	0	0	0	1	1	x	x	x	x	x	x	x	x	x	x

PC	0x0009

R0	0x00
R1	0x06

R2	0x04
R3	0x03

Z

Fetched data	0x61
Decoded Instruction	

Clk 15

Fetch and increment PC

Program Execution

Processor Z

ADDRESS																DATA									
Hex				Binary												Binary								Hex	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	1	0	9	2
0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	1	0	1	1	8	B
0	0	0	2	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	2	0
0	0	0	3	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	1	8	1
0	0	0	4	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	4
0	0	0	5	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0	0	0	0	0	0	4	0
0	0	0	6	0	0	0	0	0	0	0	0	0	1	1	0	1	1	1	1	0	1	0	0	F	4
0	0	0	7	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	1	1	1	1	8	F
0	0	0	8	0	0	0	0	0	0	0	0	1	0	0	0	0	1	1	0	0	0	0	1	6	1
0	4	0	0	0	0	0	0	0	1	0	0	0	0	0	0	x	x	x	x	x	x	x	x	x	x
0	4	0	1	0	0	0	0	0	1	0	0	0	0	0	1	x	x	x	x	x	x	x	x	x	x
0	4	0	2	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	1	1	0	3
0	4	0	3	0	0	0	0	0	1	0	0	0	0	1	1	x	x	x	x	x	x	x	x	x	x

PC	0x0009

R0	0x00
R1	0x06

R2	0x04
R3	0x03

Z

Fetched data	0x61
Decoded Instruction	
011 000 01	
Stz R1	

Clk 15
Decode

Assembly instruction mnemonic		Machine Code (binary)	Action
ldz	reg	001 000 rr	Load register from memory location pointed to by register Z
ldi	reg, vv	10 vvvv rr	Load immediate register with value v
stz	reg	011 000 rr	Store register to memory location pointed to by register Z
add	regd, regs	000 0 rd rs	Add regd to regs and place result in regd (regd ← regd + regs)
dec	reg	010 000 rr	Decrement the contents of register
bnz	reg, offset	11 vvvv rr	If the value in reg is not zero, branch to the location given by PC + offset (vvvv)

Program Execution

Processor Z

ADDRESS																DATA									
Hex				Binary												Binary								Hex	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	1	0	9	2
0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	1	0	1	1	8	B
0	0	0	2	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	2	0
0	0	0	3	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	1	8	1
0	0	0	4	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	4
0	0	0	5	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0	0	0	0	0	0	4	0
0	0	0	6	0	0	0	0	0	0	0	0	0	1	1	0	1	1	1	1	0	1	0	0	F	4
0	0	0	7	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	1	1	1	1	8	F
0	0	0	8	0	0	0	0	0	0	0	0	1	0	0	0	0	1	1	0	0	0	0	1	6	1
0	4	0	0	0	0	0	0	0	1	0	0	0	0	0	0	x	x	x	x	x	x	x	x	x	x
0	4	0	1	0	0	0	0	0	1	0	0	0	0	0	1	x	x	x	x	x	x	x	x	x	x
0	4	0	2	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	1	1	0	3
0	4	0	3	0	0	0	0	0	1	0	0	0	0	1	1	0	0	0	0	0	1	1	0	0	6

PC	0x0009

R0	0x00
R1	0x06

R2	0x04
R3	0x03

Z

Fetched data	0x61
Decoded Instruction	
011 000 01	
Stz R1	

Clk 15
Execute