LC-3

memory location 2^{16}

Mar width = 16 bits

MDR width = 16 bits

Instruction Cycle

Fetch - gets next instruction.

Decode - examines instruction.

Evaluate Address - computes address of memory location

Fetch Operands - obtains source operands

Execute - carries out instruction

Store Result - result written

Add

bit 5 = 0 = register

= 1 = immediate.

Input from keyboard

keyboard data register - KDR

keyboard status register - KSR

Bit 15 = ready bit

1 = disabled

0 = not disabled

- von neumann machine data and instructions reside in memory
- Isa specifies items including opcodes and addressing modes.
- Mar holds the number of addresses in memory.
- immediate instructions have one of their operands in the instruction itself.
- Halt is a trap instruction
- Lea modifies condition bits
- .ORig is a directive, not an instruction
- everytime a subroutine is called, a new stack frame is generated.
- LC3 uses special memory mapped I/O to communicate with IO Devices.

Assember directives

- .Fill allocates 1 word of space and fills with value of N
- .BLKW allocates n ~~characters plus~~ memory locations that are consecutive
- .Stringz allocates n characters plus 1 that are consecutive.
- .End tells assembler where the program ends

output to monitor

display data register = DDR

display status register = DSR

bit 15 = ready bit

0 = disabled

1 = not disabled

- the location counter is initialized using the .ORIG directive, as long as its not the PC
- 256 unique trap services routines

Addressing Modes -

PC-Relative - LD, ST

9 bit offset relative to PC * add 1 for PC

Indirect - LDI, STI

contains address to be loaded

Base + offset - LDR, STR

Add 6 bit @ 2's compliment value

immediate - Lea

only with Lea. add incremented PC to last 8 bits

register - and

add contents of register.

Assembly Process

1. assembly file writing
2. 1st Pass - creates a symbol table for labels or names
3. 2nd Pass - uses contents of symbol table when goin through
4. Executable Object file

Asynchronous - different speeds

Synchronous - same speeds.

Decimal to binary

10.625

10 / 2 = 5 R 0

5 / 2 = 2 R 1

2 / 2 = 1 R 0

1 / 2 = 0 R 1

10 = 1010

.625 x 2 = 1.250

.250 x 2 = 0.5

.5 x 2 = 1.0

.625 = 101

10.625 = 1010.101

floating Point IEEE

1010.101 = $(-1)^0 \times 1.010101 \times 2$

3 = E - 127

E = 130

0 10000010 01010100000000000000

addressibility - # of bits stored in each memory location

Address space - total # of uniquely identifiable locations.

Range of values

2's compliment

min = -2^{16-1} 16 is # of digits

max = $2^{16-1} - 1$

Sample Program

```
        .Orig  x3200
        LD R0, Num1      x3200
        LDI R1, num2     x3201
        Lea R2, num0     x3202
        LDR R3, R2, #0   x3203
        Halt             x3204
Num1    .Fill x3301      x3205
num2    .Fill x 3302     x3206
num0    .Fill x 3300     x3207
        .End             x3208
```

x3300	AE30
x3301	10CF
x3302	5390
x3303	FF00

R0 = x3301 relative
R1 = x5390 indirect
R2 = x3207 immediate
R3 = x3300 Base + off

Table

0000 - 0
0001 - 1
0010 - 2
0011 - 3
0100 - 4
0101 - 5
0110 - 6
0111 - 7
1000 - 8
1001 - 9
1010 - A
1011 - B
1100 - C
1101 - D
1110 - E
1111 - F

this not Digits

Levels of transformation

problem
Algorithms
Language
ISA
micro arch
logic circuit
Devices

Device Register Assignments

xFE00 - Keyboard Status register
xFE02 - Keyboard data register
xFE04 - Display Status register
xFE06 - Display data register
xFFFE - machine control register

The Trap Mechanism

1. set of service routines
2. Table of starting addresses
3. The trap instruction
4. Linkage

Stack

- Stack pointer keeps track of location of last element put onto the stack.
- attempting to pop items that have not previously been pushed is an underflow,
- overflow when attempting to push values when there is no more room.

imm5 = 5 bit 2's compliment value.
PC offset 9 = 9 bit 2's compliment value.
PC offset 11 = 11 bit 2's compliment value.
trapvect8 = zero extend 8 bits unsigned.

Add - bit[5] = 1 then immediate
Not / = 0 then register.

- the result of the linker is the executable.

operate instructions - Add, and, Not
Data movement instructions - LD, LDI, LDR, ST, STI, STR, LEA
control instructions - BR, JSR, JSRR, JMP, RTT, Trap

LD R4, value
R4 ← mem[value]

LDI R4, onemore
R4 ← mem[mem[onemo

LDR R4, R2, #-5
R4 ← mem[R2-5]

LEA R4, Target
R4 ← address of target

ST R4, HERE
mem[HERE] ← R4

STI R4, Not-Here
mem[mem[Not-Here]] ←

STR R4, R2, #5
mem[R2+5] ← R4