CS 61C: Great Ideas in Computer Architecture (Machine Structures)

Instructors:
Randy H. Katz
David A. Patterson
http://inst.eecs.Berkeley.edu/~cs61c/fa10

Agenda

- Review
- Communicating with People
- Administrivia
- Technology Break
- How Computers Make Decisions
- Summary

Review from Last Lecture

- Operands are limited in instruction sets: registers
  - 32 registers in MIPS
  - 2nd Design Principle: Smaller is faster
  - Each 32 bits wide ("word")
  - They are 100-500 times faster than word in memory
- Constants also operands
  - 3rd Design Principle : Make the common case fast.
  - Variation of instructions: addi, multi, divi,
- Computers do logical operations as well as arithmetic
  - and, or, shift left (sll), shift right (srl), andi, ori,
- Characters: 8-bit ASCII and 16-bit Unicode
- From Java to MIPS instructions called compiling
  - Compile-time (before program run) vs. run-time

Review: Memory Addresses are in Bytes

- Lots of data is smaller than 32 bits but rarely smaller than 8 bits, so works fine if everything is a multiple of 8 bits
- 8 bit item is called a byte (1 word = 4 bytes)
- Memory addresses are really in bytes, not words
- Word addresses are 4 bytes apart
  - Word address is same as leftmost byte

…	…	…	…
12	13	14	15
8	9	10	11
4	5	6	7
0	1	2	3

Strings vs. Characters

- Usually want a long sequence of characters, called a *string*
- What ways could we represent a variable length sequence of characters?

C vs. Java

- So far in arithmetic expressions, no significant difference between C and Java
- Differ starting with symbols and strings
- C: 8-bit ASCII, define strings with end of string character NUL (0 in ASCII)
- Java: 16-bit UNICODE, first entry gives length of string

Strings

- "Cal" in ASCII in C; How many bytes?
- Using 1 integer per byte, what does it look like?

ASCII value	Char-acter	ASCII value	Char-acter	ASCII value	Char-acter	ASCII value	Char-acter	ASCII value	Char-acter	ASCII value	Char-acter
32	space	48	0	64	@	80	P	96	`	112	p
33	!	49	1	65	A	81	Q	97	a	113	q
34	"	50	2	66	B	82	R	98	b	114	r
35	#	51	3	67	C	83	S	99	c	115	s
36	$	52	4	68	D	84	T	100	d	116	t
37	%	53	5	69	E	85	U	101	e	117	u
38	&	54	6	70	F	86	V	102	f	118	v
39	'	55	7	71	G	87	W	103	g	119	w
40	(	56	8	72	H	88	X	104	h	120	x
41	)	57	9	73	I	89	Y	105	i	121	y
42	*	58	:	74	J	90	Z	106	j	122	z
43	+	59	;	75	K	91	[	107	k	123	{
44	,	60	<	76	L	92	\	108	l	124	|
45	-	61	=	77	M	93	]	109	m	125	}
46	.	62	>	78	N	94	^	110	n	126	~
47	/	63	?	79	O	95	_	111	o	127	DEL

Strings

- "Cal" in Unicode in Java; How many bytes?
- Using 1 integer per byte, what does it look like? (For Latin alphabet, 1st byte is 0, 2nd byte is ASCII)

ASCII value	Char-acter	ASCII value	Char-acter	ASCII value	Char-acter	ASCII value	Char-acter	ASCII value	Char-acter	ASCII value	Char-acter
32	space	48	0	64	@	80	P	96	`	112	p
33	!	49	1	65	A	81	Q	97	a	113	q
34	"	50	2	66	B	82	R	98	b	114	r
35	#	51	3	67	C	83	S	99	c	115	s
36	$	52	4	68	D	84	T	100	d	116	t
37	%	53	5	69	E	85	U	101	e	117	u
38	&	54	6	70	F	86	V	102	f	118	v
39	'	55	7	71	G	87	W	103	g	119	w
40	(	56	8	72	H	88	X	104	h	120	x
41	)	57	9	73	I	89	Y	105	i	121	y
42	*	58	:	74	J	90	Z	106	j	122	z
43	+	59	;	75	K	91	[	107	k	123	{
44	,	60	<	76	L	92	\	108	l	124	|
45	-	61	=	77	M	93	]	109	m	125	}
46	.	62	>	78	N	94	^	110	n	126	~
47	/	63	?	79	O	95	_	111	o	127	DEL

Instructions to support characters and strings in C

- Load a word, use andi to isolate byte

```
lw $s0, 0($s1)
andi $s0,$s0, 255    # Zero everything but last 8 bits
```

- Since design rule 3 is "Make the Common Case Fast," and many programs use text, MIPS has *load byte* instruction (*lb*)

```
lb $s0, 0($s1)
```

- Also *storage byte* instruction (*sb*)

Instructions to support characters and strings in Java

- Load a word, use andi to isolate half of word

```
lw $s0, 0($s1)
andi $s0,$s0, 65535 # Zero everything but last 16 bits
```

- Since design rule 3 is "Make the Common Case Fast", and many programs use text, MIPS has *load halfword* instruction (*lh*)

```
lh $s0, 0($s1)
```

- Also *storage byte* instruction (*sh*)

How Computers Make Decisions

- Based on computation, do something different
- In programming languages, an if-statement
- Sometimes combined with gotos and labels
- In MIPS the if-statement instruction is

```
beq register1, register2, L1
```

means go to statement labeled L1
if value in register1 = value in register2

- beq stands for *branch if equal*
- Other instruction: bne for *branch if not equal*

Making Decisions in C or Java

if (i == j) f = g + h; else f = g – h;

- If false, skip over "then" part to "else" part
  => use conditional branch bne
- Otherwise, (its true) do "then" part and skip over "else" part => need an always branch instruction ("*unconditional branch*")
- MIPS name for this instruction is *jump (j)*

Making Decisions in MIPS

if (i == j) f = g + h; else f = g – h;

- f => $s0, g => $s1, h => $s2, i => $s3, j => $s4
- If false, skip “then” part to “else” part
- Otherwise, (its true) do “then” part and skip over “else” part

Making Decisions in MIPS

if (i == j) f = g + h; else f = g – h;

- f => $s0, g => $s1, h => $s2, i => $s3, j => $s4
- If false, skip “then” part to “else” part
- Otherwise, (its true) do “then” part and skip over “else” part

```
        bne $s3,$s4,Else  # go to Else part if i ≠ j
        add $s0,$s1,$s2   # f = g + h (Then part)
        j Exit            # go to Exit
Else:   sub $s0,$s1,$s2   # f = g – h (Else part)
Exit:
```

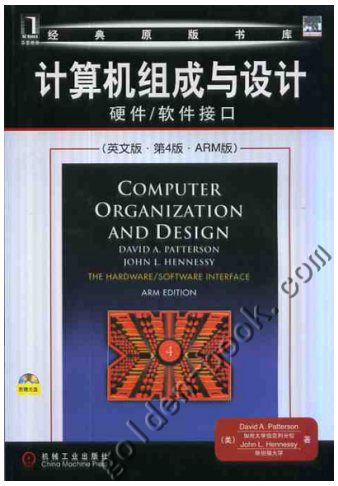
The Secret to Getting Good Grades

- Grad student said he figured finally it out
  - (Mike Dahlin, now Professor at UT Texas)
- What is the secret?
- Do assigned reading night before, so that get more value from lecture

Administrivia

- You can drop in the dropbox for 61C in 283 Soda
- We'll try to get electronic submission set up by tomorrow (Thursday) and post on web site and email via Google Group
- 30 computers in room, 36 students assigned to several labs, bring your laptop
- Can do labs in pairs
- Small reward for checkoff in 1st hour
- If class account problems, come up at break

ARM (Asian) edition is NOT OK (wrong instruction set, wrong appendices)



Ask for your money back if they guaranteed it's the same as US Edition, since its different

When is Midterm, Final?

- To reduce time pressure, 3 hours for 1.5 hour midterm
- **Midterm Exam Wednesday October 6, 6 – 9PM, Pimental 1**
- **Final Exam Monday December 13, 8 – 11AM, TBD**

61C In The News: "Intel acquisition indicates shift away from PCs" 8/31

"Intel, the reigning champion of PC processors, is increasingly preparing for life in the post-PC era. While the company owns about 80 percent of the market for computer processors, it is increasingly showing signs that it plans to compete vigorously in the market for smaller, more nimble computing devices such as smart phones, tablets and e-readers.

,,,

"As more devices compute and connect to the Internet, we are committed to positioning Intel to take advantage of the growth potential in every computing segment, from laptops to handhelds and beyond," said Intel CEO Paul Otellini….

Forrester projects this year will be the first in which sales of "post-PC" devices - smart phones, tablets and e-readers - eclipse PC sales in the United States [RISC vs. x86]. There will be 66 million post-PC devices sold this year, compared with 55 million PCs, according to Forrester's forecast. By 2015, the gap will widen to 93 million post-PC devices to 69 million PCs.

The whole PC industry is going through a similar transition, some more quickly than others. Apple made the move in 2007 with the release of the iPhone. It dropped the word "computer" from its title that year, and earlier this year, CEO Steve Jobs declared that Apple was essentially a mobile company.

Adam Gross, said the launch of the iPad has created a clear demarcation between the PC era and the devices era. Now, with the power of cloud computing, where you store information in remote servers, and the growing desire to be mobile, the dominant usage model of PCs is getting upended, he said.

Gross said. "But the minute you introduce new devices like the iPhone, Android phones or the iPad, it doesn't make sense to have one thing be the locus of the computing environment anymore."

For companies like Intel, there is no guarantee that their transition will prove successful. Intel must still make its processor more efficient and win over device manufacturers, who have many low-priced choices."

The Rules
(and we really mean it!)

Agenda

- Review
- Communicating with People
- Administrivia + Review of the secret to getting good grades at Berkeley
- Technology Break
- How Computers Make Decisions
- Summary

How do other comparisons?

- To keep things simple, only = and ≠ branches
- MIPS set on less than (slt) sets a register to 1 if a register is less than another, to 0 otherwise

```
slt $t0,$s3,$s4 # t0=1 if s3<s4
```

- A branch on less than takes 1 more instruction

```
bne $t0,$zero,L # branch if s3<s4
```

- ≤,>,≥ by swapping operands and/or using beq

Loops

```
while (save[i] == k)
        i += 1;
```

- Load save[i] into a temporary register
- Must first multiply index i by 4 due to byte addressing
- Then add to base of save[] and load
- If not equal to k, exit the loop
- Increment i
- Branch back to instruction at end of the loop

Loops

```
while (save[i] == k)
        i += 1;
```

- i => $s3, k => $s5, base of array save is in $s6

Loops: Why not this way? (5 vs. 6 instructions/iteration)

```
while (save[i] == k)
        i += 1;
```

- i => $s3, k => $s5, base of array save is in $s6

```
Loop:  multi $t1,$s3,4     # Temp reg $t1 = i * 4
       add $t1,$t1,$s6     # $t1 = address of save[i]
       lw $t0,0($t1)       # Temp reg $t0 = save[i]
       addi $s3,$s3,1      # i = i + 1
       beq $t0,$s5, Loop   # go to Loop if save[i] = k
Exit:
```


Putting it all together: C string copy

- Built in C library called string copy (strcpy)

```
i = 0;
while( (x[i] = y[i]) != '\0')
                    /* copy & test byte */
   i += 1;
```

- What does MIPS code look like for it?
- Assume `i` in $s0, source string address in `$s1`, destination string address in `$s2`

String Copy in C

```
i = 0;
while( (x[i] = y[i]) != '\0') /* copy & test byte */
    i += 1;
```

i => $s0, source address in $s1, destination in $s2

```
                                # i = 0
Loop:                           # address of y[i] in $t1
                                # $t2 = y[i]
                                # address of x[i] in $t3
                                # x[i] = y[i]
                                # if y[i] == 0, go to Exit
                                # i = i + 1
                                # go to Loop
Exit:
```


Summary

- People need text as well as numbers
- C: 8-bit ASCII, Java: 16-bit Unicode
- load byte (lb), store byte (sb), load halfword (lh), store halfword (sh) support characters
- Decisions in C and Java (if, while, repeat, for, …) via conditional branch:
  - Branch equal (beq) register1, register2, address
  - Branch not equal (bne) register1, register2, addr.
- Unconditional Branch (jump or j) too
- C pointers = MIPS addresses