COMP1521 25T3

Week 3 Lecture 2

MIPS FUNctions

Announcements

- Help Session Schedule is out
  - COMP1521 25T3 — COMP1521 Help Sessions
  - BYOD as they are not in labs

- Assignment 1 out soon!

- Labour day public holiday Monday next week
  - Ask tutor permission before joining another TLB

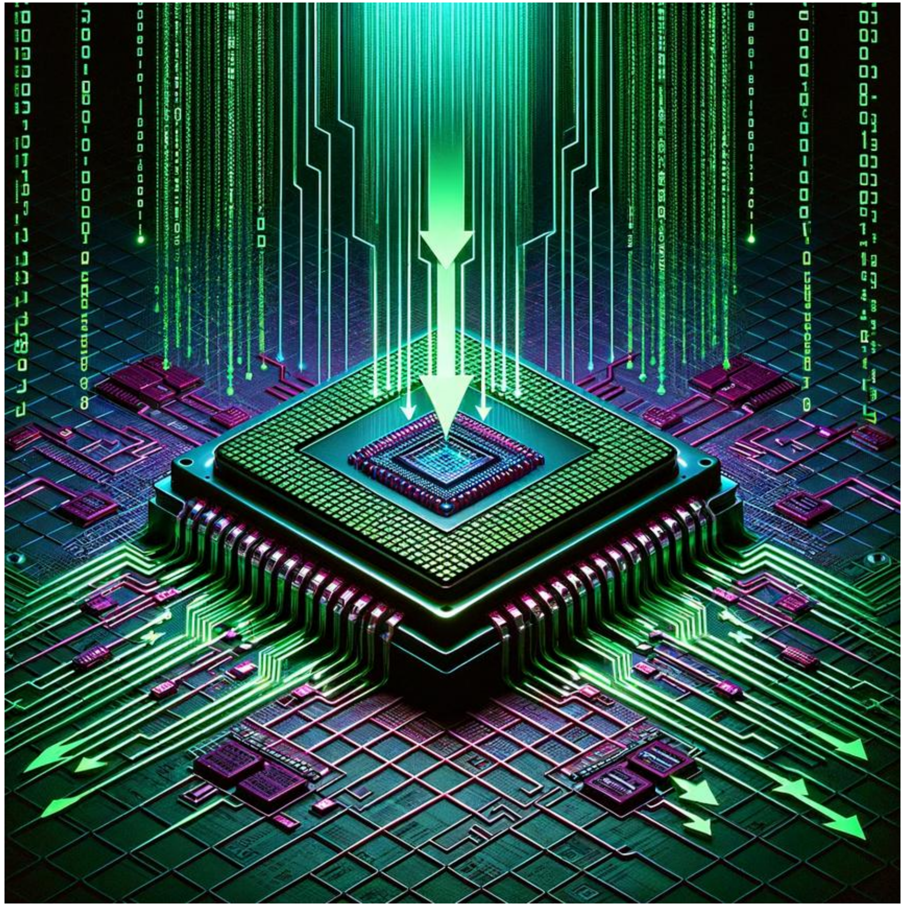
Reminder: First Weekly Test Out Tomorrow

Released: Thursday 3pm
Time limit: 1 hour
Due: Thursday Week 4 at 3pm. (And then another test comes out)
Submitted via **give**
You can get 50% max for questions submitted after the hour is up

Topic for week 3 test: MIPS basics, control.
Self-enforced exam conditions!
You can use mips documentation

Today’s Lecture

- Recap functions
  - Calling functions
  - Stacking registers
  - MIPS ABI
- More function examples
- A MIPS application;
  Putting it all together



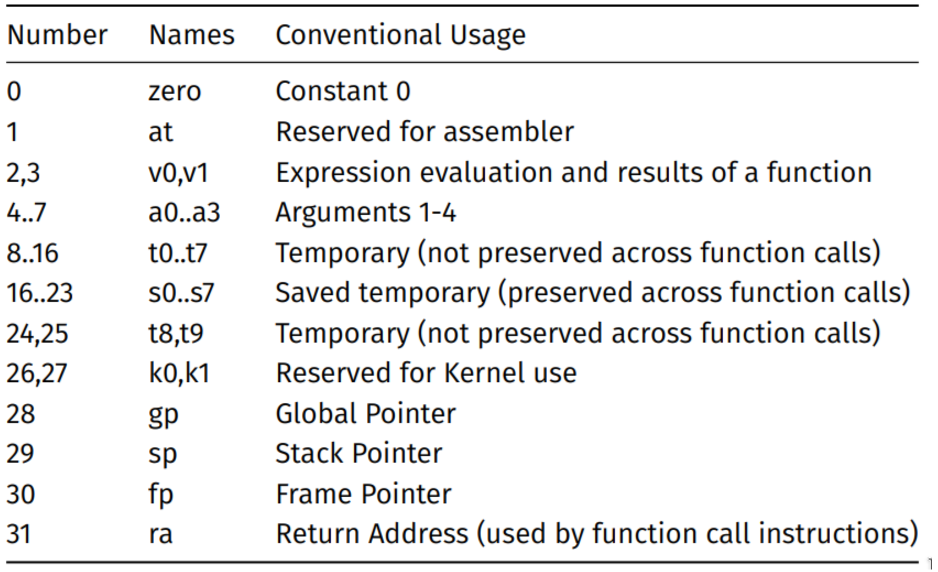
Functions - a summary

- Functions are named pieces of code (**labels**)
    - Which you can call (`jal`)
    - Which you can (optionally) supply arguments (`$a0 - $a3`)
    - Perform computations using those arguments (`add/mul/etc`)
    - And return a value to a caller (`$v0`)

MIPS ABI: Summary

- `$t` registers are free real estate
  - So we must assume that other functions destroy them
- A function must restore the original values of `$sp`, `$fp`, `$s0`..`$s7`
  - So we can assume that any function we call leaves these registers unchanged
- Functions need to preserve `$ra` if they overwrite it (e.g. using `jal`)
  - Otherwise, our function will lose track of where to return to
- `$a0`..`$a3` contain our arguments -
  - these are also not preserved by callees (like `$t`)
- `$v0` contains the return value

MIPS ABI: Summary



Number	Names	Conventional Usage
0	zero	Constant 0
1	at	Reserved for assembler
2,3	v0,v1	Expression evaluation and results of a function
4..7	a0..a3	Arguments 1-4
8..16	t0..t7	Temporary (not preserved across function calls)
16..23	s0..s7	Saved temporary (preserved across function calls)
24,25	t8,t9	Temporary (not preserved across function calls)
26,27	k0,k1	Reserved for Kernel use
28	gp	Global Pointer
29	sp	Stack Pointer
30	fp	Frame Pointer
31	ra	Return Address (used by function call instructions)

Function Skeleton

```
func:
        # [header comment]
func__prologue:
        begin
        push    $ra
        push    $s0
        push    $s1

func__body:
        # do stuff

        li      $a0, 42
        jal     foo             # foo(42)

        # foo return val in $v0

        # at the end of the function
func__epilogue:
        pop     $s1
        pop     $s0
        pop     $ra
        end

        jr      $ra
```

Implement this: return value

```c
int f(int x);

int main(void) {
  printf("calling function f\n");
  int result = f(22);
  printf("back from function f\n");
  printf("%d", result);
  putchar('\n');
  return 0;
}
```

```c
int f(int x) {
  printf("in function f\n");
  printf("%d", x);
  putchar('\n');
  x = x + 1;
  return x;
}
```

Recap Exercises

function_example_broken.s

sum_to.c

sum_to_r.c

MIPS Pizzeria Application

MIPS Pizzeria: Data Types

```c
// Written by Hammond Pearce
include <stdio.h>

struct pizza_t {
    char size[10];
    int price_cents;
};

struct pizza_t pizza_options[3] = {
    {"small", 300},
    {"medium", 550},
    {"large", 800}
};
```

MIPS Pizzeria: Main

```
int main(void) {
    printf("The available pizza options are:\n");
    for (int i = 0; i < 3; i++) {
        increase_price(&pizza_options[i], 100);
        print_pizza_t(&pizza_options[i]);
    }
    return 0;
}
```

MIPS Pizzeria: Functions

```c
void print_pizza_t(struct pizza_t *pizza) {
    printf("Size: %s, ", pizza->size);
    printf("price: %d cents\n", pizza->price_cents);
}

void increase_price(struct pizza_t *pizza, int increase_cents) {
    pizza->price_cents += increase_cents;
}
```

That's all! No more MIPS!

Ok.. A little more MIPS..

Assignment 1

Out now!

Don’t forget before jumping into MIPS

- For each function
  - Simplify function in C
  - Compile and rerun the program to check it still works
- Don’t change everything at once without testing!

Writing Code in MIPS

- Plan register usage
- Style - consistent naming of labels
- Indentation
- Comments - equivalent line of C Code for MIPS code.

Next Week

Integer Representation

Bitwise Operations

Reach Out

Content Related Questions:
[Forum]

Admin related Questions email:
cs1521@cse.unsw.edu.au

Student Support | I Need Help With...

My Feelings and Mental Health Managing Low Mood, Unusual Feelings & Depression	**Mental Health Connect**	student.unsw.edu.au/**counselling** Telehealth	**In Australia Call Afterhours UNSW Mental Health Support Line**	1300 787 026 5pm-9am
	Mind HUB	student.unsw.edu.au/**mind-hub** Online Self-Help Resources	**Outside Australia Afterhours 24-hour Medibank Hotline**	+61 (2) 8905 0307
Uni and Life Pressures Stress, Financial, Visas, Accommodation & More	**Student Support Indigenous Student Support**	— student.unsw.edu.au/**advisors**		
Reporting Sexual Assault/Harassment	**Equity Diversity and Inclusion (EDI)**	— edi.unsw.edu.au/**sexual-misconduct**		
Educational Adjustments To Manage my Studies and Disability / Health Condition	**Equitable Learning Service (ELS)**	— student.unsw.edu.au/**els**		
Academic and Study Skills	**Academic Language Skills**	— student.unsw.edu.au/**skills**		
Special Consideration Because Life Impacts our Studies and Exams	Special Consideration	— student.unsw.edu.au/**special-consideration**		