COMP1521 25T2

Week 2 Lecture 2

MIPS Data and Memory

Adapted from Angela Finlayson, Abiram Nadarajah, Hammond Pearce, Andrew Taylor and John Shepherd's slides

C revision session and Help Sessions

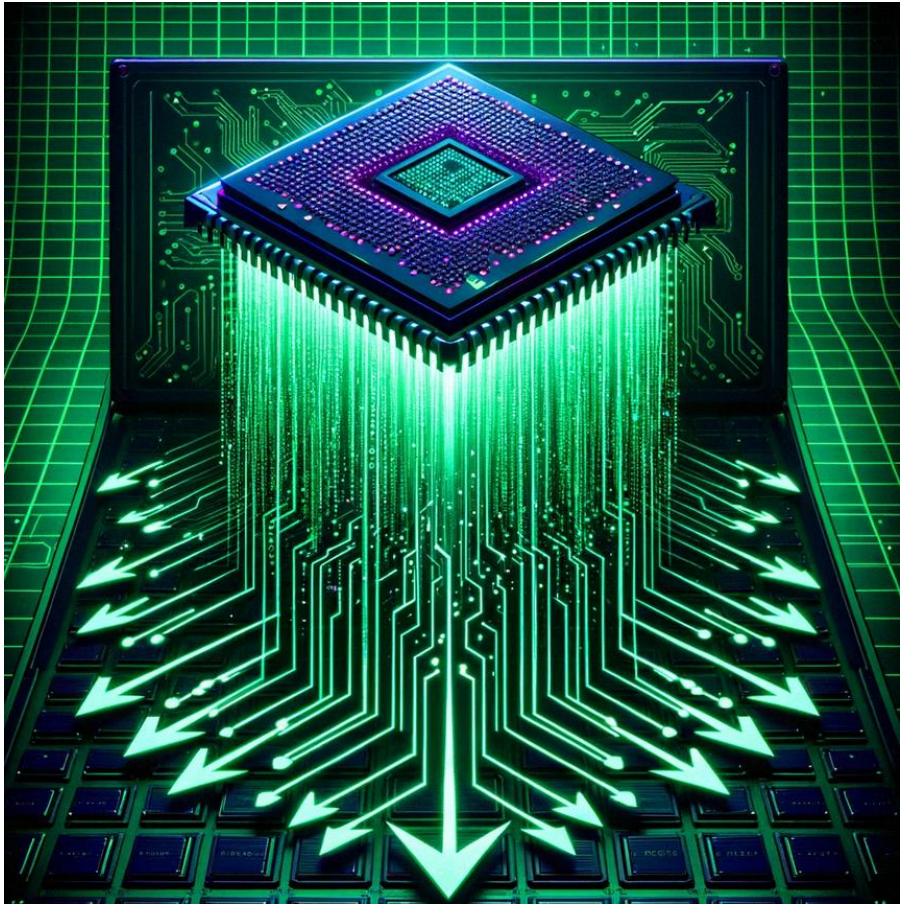
- C revision session: Thu (TOMORROW!) 10am-12pm
  Where? Blackboard collaborate on Moodle.

- More info on the forum under announcements
  - Week 2 Revision Lab + some resources for learning/revising C - Announcements - COMP1521

- Help Session Schedule coming out soon
  - Sessions starting on Monday!

Today's Lecture

- Recap Mondays lecture
  - Data and Memory
    - Global variables
- 1D arrays
- 2D arrays
- C structs



Recap: Accessing Memory

- Loading data:
  - To perform computations, data must be transferred from memory into the CPU registers

- Storing data:
  - Modified data must be written back from the CPU registers to memory

Recap Exercise

```c
char letter = 'B';

int main(void) {
    letter--;
    printf("%c", letter);
    putchar('\n');
    return 0;
}
```

Mipsy assembler directives

```
.text                   # following instructions placed in text segment
.data                   # following objects placed in data segment


a: .space 18            # int8_t a[18];
.align 2                # align next object on 4-byte addr
i: .word 42             # int32_t i = 42;
v: .word 1,3,5          # int32_t v[3] = {1,3,5};
h: .half 2,4,6          # int16_t h[3] = {2,4,6};
b: .byte 7:5            # int8_t b[5] = {7,7,7,7,7};
f: .float 3.14          # float f = 3.14;
s: .asciiz "abc"        # char s[4] {'a','b','c','\0'};
t: .ascii "abc"         # char t[3] {'a','b','c'};
```

Initialising a global arrays

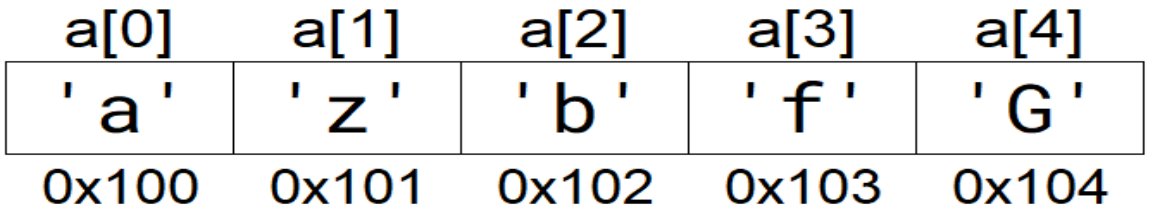
How can we access elements?

How can we loop through the arrays?

```
vec:   .space 40                    # int vec[10] or char vec[40]
nums:  .word 1, 3, 5, 7, 9          # int nums[6] = {1,3,5,7,9}
nums2:  .word 1:4                   # int nums[4] = {1,1,1,1}
str:   .byte 'a', 'b', 'c', '\0'    # char str[] = {'a','b','c','\0'}
str2:  .asciiz "abc"                # char str2[] = "abc"
chars:  .ascii "abc"                # char str2[] = {'a','b','c'}
```

Arrays of 1 byte elements (array.c demo)

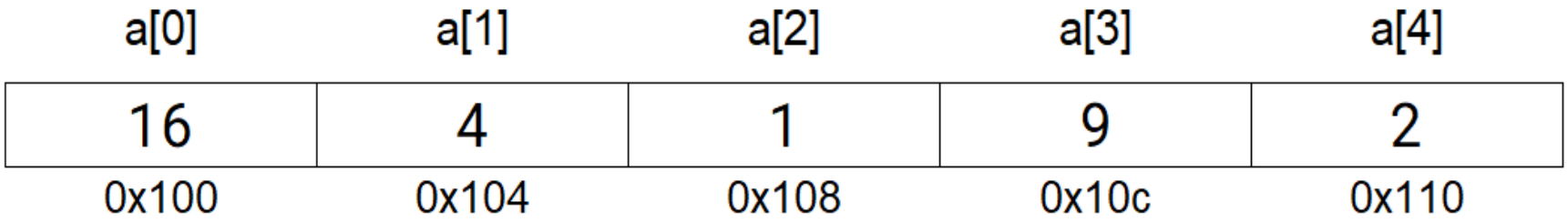
char a[5] = {'a', 'z', 'b', 'f', 'G'};



- If we have the address of the start of the array:
  - How can I work out the address of the a[3]?
  - How can I work out the address of the a[i]?

Arrays of 4 byte elements

int a[5] = {16, 4, 1, 9, 2};



- If we have the address of the start of the array:
  - How can I work out the address of the a[3]?
  - How can I work out the address of the a[i]?

Address of Array Elements

char array: address of a[i] = address of a + i

integer array: address of a[i] = address of a + (i * 4)

In general:

address of element = address of array + index * sizeof(element)

MIPS array coding examples (array_bytes.s)

```
char a[] = {'a', 'z', 'b', 'f', 'G'};
int main(void) {
    for (int i = 0; i < sizeof(a); i++) {
        printf("%c", a[i]);
        putchar(' ');
    }
    putchar('\n');
    return 0;
}
```

MIPS array coding examples (array_words.s)

```
int a[] = {3, 9, 7, 3, 1};
int main(void) {
    for (int i = 0; i < sizeof(a)/sizeof(*a); i++) {
        printf("%d", a[i]);
        putchar(' ');
    }
    putchar('\n');
    return 0;
}
```

Pointer Arithmetic in C

In C adding 1 to a pointer increases it by the **sizeof** the type it points to!

This makes it easy to use a pointer to iterate through an array!

```
char   *p = 0x6060;  p++;  // (p == 0x6061)
int    *q = 0x6060;  q++;  // (q == 0x6064)
double *r = 0x6060;  r++;  // (r == 0x6068)
```

In MIPS we have to take the size into account ourselves!

Pointer Arithmetic in C

Rule | Explanation

Pointer Arithmetic

Avoid pointer Arithmetic.

Use array indices instead.

Experienced programmers use pointer arithmetic to produce succinct idiomatic code.

Novice programmers confuse themselves by trying to use pointer arithmetic. Any code using pointer arithmetic can also be written using array indices. Use array indices unless you are confident in your programming ability and are sure it is produces more readable code then array indices.

Pointer Arithmetic

```
int a[5] = {3, 9, 7, 3, 1};
int main(void) {
    int* p = &a[0];
    int* q = &a[4];
    while (p <= q) {
        printf("%d", *p);
        putchar(' ');
        p++;
    }
    putchar('\n');
    return 0;
}
```

2D Arrays in MIPS

	0	1	2	3
0	a	b	c	d
1	e	f	g	h
2	i	j	k	l

< col

^ row

RAM is really just a 1D array.
A 2D array is really represented in memory with each row next to each other.

We need to map our 2 indexes to the appropriate offset

a	b	c	d	e	f	g	h	i	j	k	l
0	1	2	3	4	5	6	7	8	9	10	11

2D Arrays in MIPS

	0	1	2	3
0	a	b	c	d
1	e	f	g	h
2	i	j	k	l

< col

^ row

Offset of start of relevant row:

(row * N_COLS) * sizeof(element)

Offset within row:

col * sizeof(element)

Total offset:

(row * N_COLS + col) * sizeof(element)

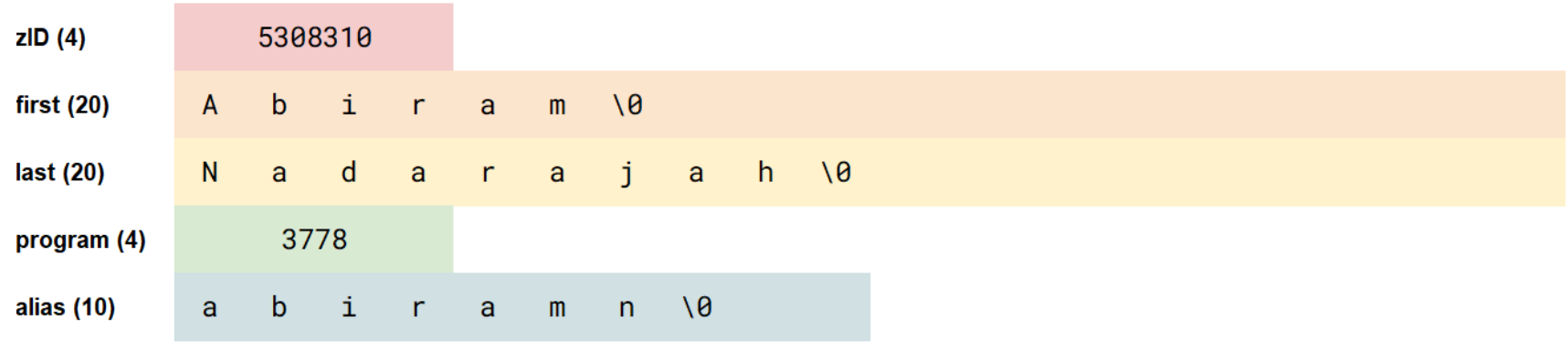
a	b	c	d	e	f	g	h	i	j	k	l
0	1	2	3	4	5	6	7	8	9	10	11

MIPS 2d array coding examples (flag.c)

cgi.cse.unsw.edu.au/~cs1521/25T2/topic/mips_data/code/flag.c

Structs

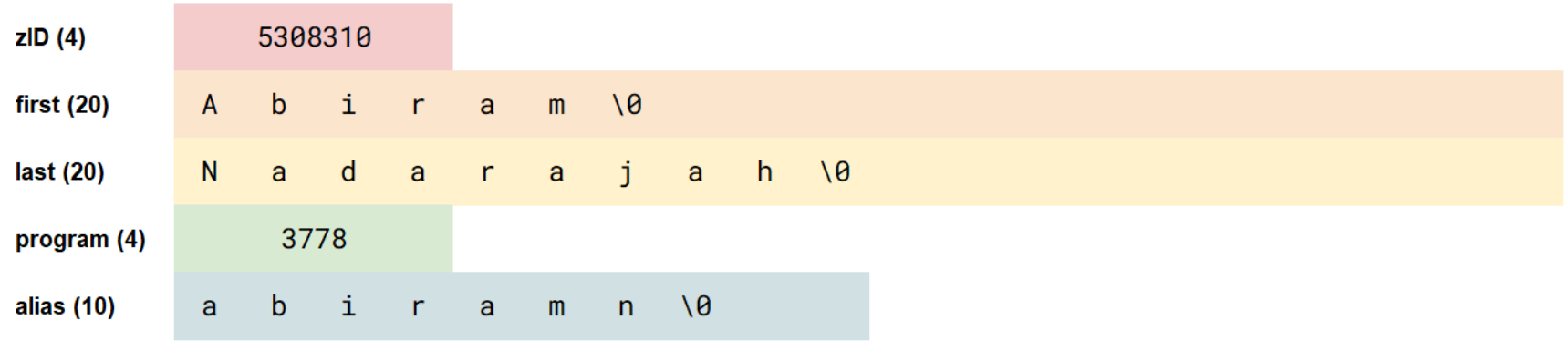
```
struct student {
    int zid;
    char first[20];
    char last[20];
    int program;
    char alias[10];
};
```



Structs

```
struct student {
    int zid;          //Offset 0
    char first[20];   //Offset 4
    char last[20];    //Offset 24
    int program;      //Offset 44
    char alias[10];   //Offset 48
};
```

structs are really just sets of variables at known offsets



What did we learn today?

- MIPS
  - recap of loading and storing data and pointers
  - arrays (1d and 2d)
  - structs
- Next lecture:
  - Functions in MIPS

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
- **Mind HUB** — student.unsw.edu.au/**mind-hub** Online Self-Help Resources

In Australia Call Afterhours UNSW Mental Health Support Line — 1300 787 026 5pm-9am

Outside Australia Afterhours 24-hour Medibank Hotline — +61 (2) 8905 0307

Uni and Life Pressures
Stress, Financial, Visas, Accommodation & More

- **Student Support** — student.unsw.edu.au/**advisors**
- **Indigenous Student Support**

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**