

CSE 420

Fundamentals of Computer Architecture

Assembly Language Programming

Prof. Michel A. Kinsy,
Instructors Mishel Paul & Nges Njungle

1

The Big Picture

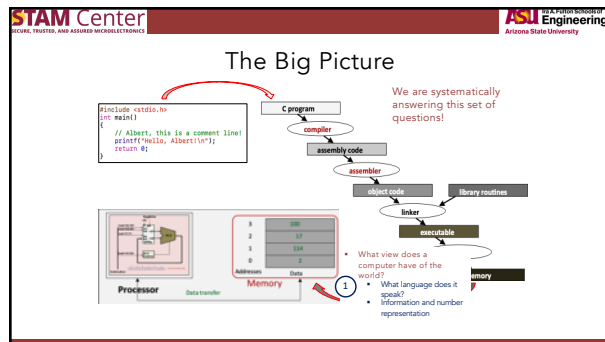
Address	Data
0	2
1	134
2	17
3	100

2

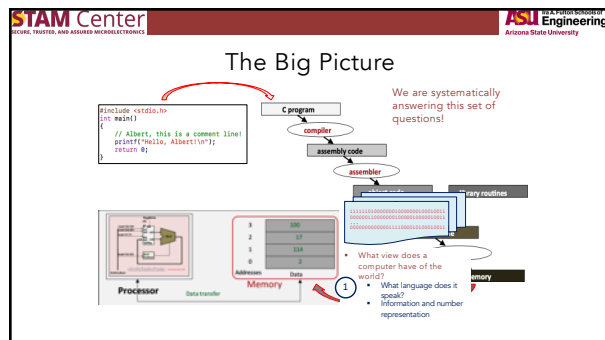
The Big Picture

Address	Data
0	2
1	134
2	17
3	100

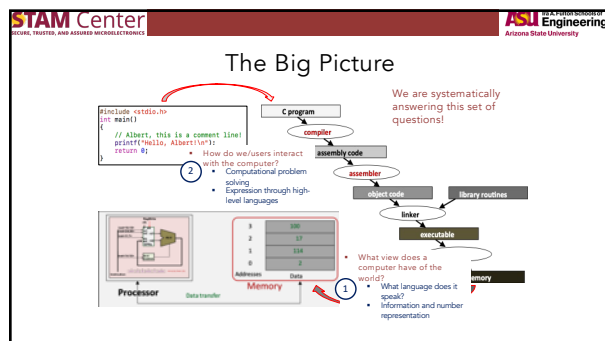
3



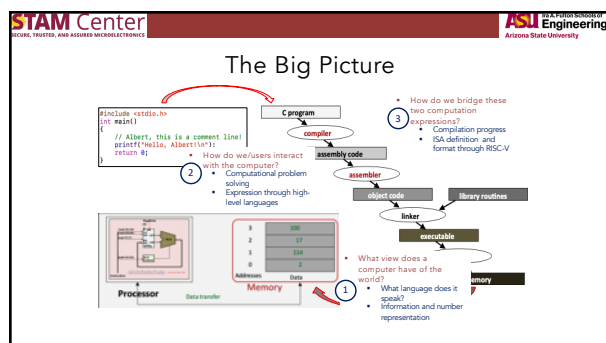
4



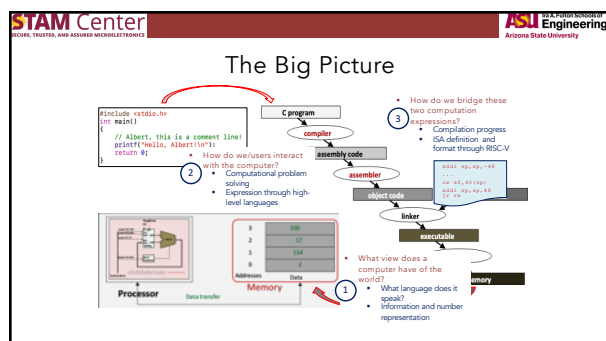
5



6



7



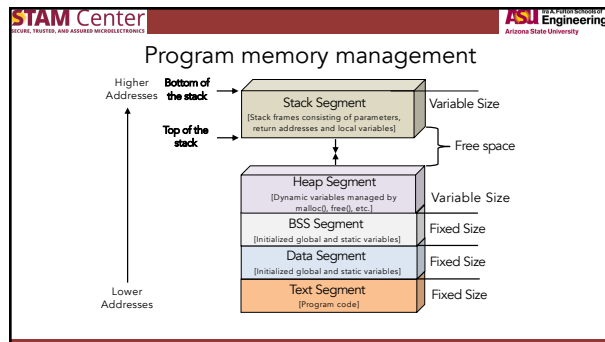
8

STAM Center Secure, Trusted, and Assured Microelectronics **ASU Engineering** Arizona State University

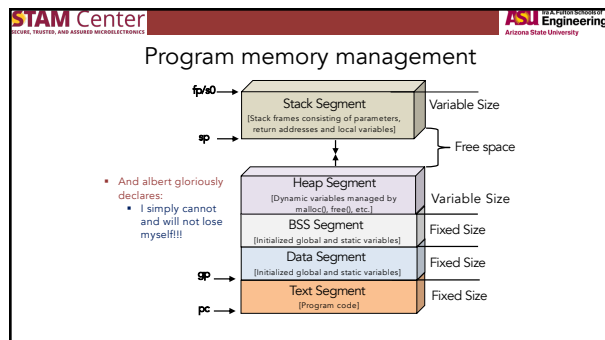
RISC-V ISA Registers

Name	Reg Number	Usage	Preserved across call?
zero	x0	The constant value 0	Yes
ra	x1	Return address	Yes
sp	x2	Stack pointer	Yes
gp	x3	Global pointer	Yes
tp	x4	Thread pointer	Yes
t0 – t2	x5 – x7	Temporary registers	No
s0/fp	x8	Frame pointer	Yes
s1	x9	Saved register	Yes
a0 – a1	x10 – x11	Function arguments/Return values	No
a2 – a7	x12 – x17	Function arguments	No

9



10



11

STAM Center **ASU Engineering**
SECURE, TRUSTED, AND ASSURED MICROELECTRONICS

RISC-V Assembly

```

int val1 = 5;
int val2;
int val3 = 2;
int val4[3] = {7,8,9};
char string1[] = "abc";
char string2[3] = {'a','b','c'};

char array1[5];
int array2[2];
short int array3[3];

int foo (int num1, int num2, int num3, int num4)
{
    int result;
    result = (num1 + num2) - (num3 + num4);
    return result;
}

void main(void) {
    array2[0] = foo(val1, val2, val3, val4[1]);
}
    
```

```

val1: word 5
val2: zero 4
val3: word 2
val4: word 7
       word 8
       word 9
string: string "abc"
string0: .byte 97
         .byte 98
         .byte 99
array1: zero 5
array2: zero 8
array3: zero 6
    
```

12

Secure, Trusted, and Assured Microelectronics

Arizona State University

RISC-V Assembly

```

val1: .word 5
val2: .zero 4
val3: .word 2
val4: .word 7
      .word 8
      .word 9
string1: .string "abc"
string2: .byte 97
        .byte 98
        .byte 99
array1: .zero 5
array2: .zero 8
array3: .zero 6
    
```

Address	Value	Address	Value
0	5	C	7
1	0	D	0
2	0	E	0
3	0	F	0
4	?	10	8
5	?	11	0
6	?	12	0
7	?	13	0
8	2	14	9
9	0	15	0
A	0	16	0
B	0	17	0

13

Secure, Trusted, and Assured Microelectronics

Arizona State University

Program memory management

The diagram illustrates the layout of program memory segments. From top to bottom, they are: Stack Segment (grows down from fp/w0), Heap Segment (grows up from sp), BSS Segment (uninitialized global and static variables), Data Segment (initialized global and static variables), and Text Segment (program code). The Stack and Heap segments are labeled as 'Variable Size', while BSS, Data, and Text are 'Fixed Size'. Arrows indicate the growth direction of the Stack and Heap segments towards each other, leaving 'Free space' in between.

14

Secure, Trusted, and Assured Microelectronics

Arizona State University

RISC-V Assembly

```

val1: .word 5
val2: .zero 4
val3: .word 2
val4: .word 7
      .word 8
      .word 9
string1: .string "abc"
string2: .byte 97
        .byte 98
        .byte 99
array1: .zero 5
array2: .zero 8
array3: .zero 6
    
```

Address	Value	Address	Value
0	5	C	7
1	0	D	0
2	0	E	0
3	0	F	0
4	?	10	8
5	?	11	0
6	?	12	0
7	?	13	0
8	2	14	9
9	0	15	0
A	0	16	0
B	0	17	0

15

STAM Center

SECURE, TRUSTED, AND ASSURED MICROELECTRONICS

ASU Engineering

Arizona State University

RISC-V Assembly

```

val1: .word 5
val2: .zero 4
val3: .word 2
val4: .word 7
      .word 8
      .word 9
string1: .string "abc"
string2:
      .byte 37
      .byte 98
      .byte 99
array1: .zero 5
array2: .zero 8
      .zero 8
    
```

Address	Value	Address	Value
8000	5	800C	7
8001	0	800D	0
8002	0	800E	0
8003	0	800F	0
8004	?	8010	8
8005	?	8011	0
8006	?	8012	0
8007	?	8013	0
8008	2	8014	9
8009	0	8015	0
800A	0	8016	0
800B	0	8017	0

Program code addresses vs. hardware addresses

16

STAM Center

SECURE, TRUSTED, AND ASSURED MICROELECTRONICS

ASU Engineering

Arizona State University

RISC-V Assembly

```

val1: .word 5
val2: .zero 4
val3: .word 2
val4: .word 7
      .word 8
      .word 9
string1: .string "abc"
string2:
      .byte 37
      .byte 98
      .byte 99
array1: .zero 5
array2: .zero 8
      .zero 8
    
```

Address	Value
8018	0x61
8019	0x62
801A	0x63
801B	0
801C	0x61
801D	0x62
801E	0x63
801F	?
8020	?
8021	?
8022	?
8023	?

Program code addresses vs. hardware addresses

17

STAM Center

SECURE, TRUSTED, AND ASSURED MICROELECTRONICS

ASU Engineering

Arizona State University

RISC-V Assembly – Let's Go All In

```

int val1 = 5;
int val2;
int val3 = 2;
int val4[3] = {7,8,9};

char string[1] = "abc";
char string2[3] = {'a','b','c'};

char array[13];
      .byte 98
      .byte 99
short int array2[3];

int foo (int num1, int num2, int num3, int num4)
{
    int result;
    result = (num1 + num2) - (num3 + num4);
    return result;
}

void main(void) {
    array2 [0] = foo(val1, val2, val3, val4[1]);
}
    
```

18



STAM Center
SECURE, TRUSTED, AND ASSURED MICROELECTRONICS



ASU Engineering
Arizona State University

RISC-V Assembly – Let's Go All In

```

int val1 = 5;
int val2;
int val3 = 2;
int val4[3] = {7,8,9};

char string[] = "abc";
char string2[3] = {'a','b','c'};

char array[15];
int array2[2];
short int array3[3];

int foo (int num1, int num2, int num3, int num4)
{
    int result;
    result = (num1 + num2) - (num3 + num4);
    return result;
}

void main(void) {
    array2 [0] = foo(val1, val2, val3, val4[1]);
}
    
```

Note that: num1, num2, num3 and num4 are a0, a1, a2, and a3, respectively.

```

foo:
add t0,a0,zero    # move num1 into t0
add t1,a1,zero    # move num2 into t1
add t2,a2,zero    # move num3 into t2

add t2,t2,a3      # t2 = num3 + num2
add t0,t0,t1      # t0 = num3 + num4

sub t1,t2,t0
# result = (num1 + num2) - (num3 + num4);

add a0,t1,zero
# result returned in a0
jr ra
    
```

19



STAM Center
SECURE, TRUSTED, AND ASSURED MICROELECTRONICS



ASU Engineering
Arizona State University

RISC-V Assembly – Let's Go All In

```

int val1 = 5;
int val2;
int val3 = 2;
int val4[3] = {7,8,9};

char string[] = "abc";
char string2[3] = {'a','b','c'};

char array[15];
int array2[2];
short int array3[3];

int foo (int num1, int num2, int num3, int num4)
{
    int result;
    result = (num1 + num2) - (num3 + num4);
    return result;
}

void main(void) {
    array2 [0] = foo(val1, val2, val3, val4[1]);
}
    
```

Silly Albert! Who are you kidding!

Note that: num1, num2, num3 and num4 are a0, a1, a2, and a3, respectively.

```

foo:
add t0,a0,zero    # move num1 into t0
add t1,a1,zero    # move num2 into t1
add t2,a2,zero    # move num3 into t2

add t2,t2,a3      # t2 = num3 + num2
add t0,t0,t1      # t0 = num3 + num4

sub t1,t2,t0
# result = (num1 + num2) - (num3 + num4);

add a0,t1,zero
# result returned in a0
jr ra
    
```

20



STAM Center
SECURE, TRUSTED, AND ASSURED MICROELECTRONICS



ASU Engineering
Arizona State University

RISC-V Assembly – Let's Go All In

```

int val1 = 5;
int val2;
int val3 = 2;
int val4[3] = {7,8,9};

char string[] = "abc";
char string2[3] = {'a','b','c'};

char array[15];
int array2[2];
short int array3[3];

int foo (int num1, int num2, int num3, int num4)
{
    int result;
    result = (num1 + num2) - (num3 + num4);
    return result;
}

void main(void) {
    array2 [0] = foo(val1, val2, val3, val4[1]);
}
    
```

```

.global val1
.section .sdata,"aw",@progbits
.align 2
.type val1,@object
.size val1,4
val1:
.word 5

.global val2
.section .sdata,"aw",@progbits
.align 2
.type val2,@object
.size val2,4
val2:
.zero 4

.global val3
.section .sdata
.align 2
.type val3,@object
.size val3,4
val3:
.word 2
    
```

21



SECURE, TRUSTED, AND ASSURED MICROELECTRONICS



Arizona State University

RISC-V Assembly – Let's Go All In

```

int val1 = 5;
int val2;
int val3 = 2;
int val4[3] = {7,8,9};

char string1[] = "abc";
char string2[3] = {'a','b','c'};

char array1[5];
int array2[2];
short int array3[3];

int foo (int num1, int num2, int num3, int num4)
{
    int result;
    result = (num1 + num2) - (num3 + num4);
    return result;
}

void main(void)
{
    array2 [0] = foo(val1, val2, val3, val4[1]);
}

```

```

.section .data,"aw",@rodata
.align 2
.type val1, @object
.size val1, 4

val1:
.word 5

.globl val2
.section .abs,"aw",@nobits
.align 2
.type val2, @object
.size val2, 4

val2:
.zero 4

.globl val3
.section .sdata
.align 2
.type val3, @object
.size val3, 4

val3:
.word 2

```

22



SECURE, TRUSTED, AND ASSURED MICROELECTRONICS



Arizona State University

RISC-V Assembly – Let's Go All In

```

int val1 = 5;
int val2;
int val3 = 2;
int val4[3] = {7,8,9};

char string1[] = "abc";
char string2[3] = {'a','b','c'};

char array1[5];
int array2[2];
short int array3[3];

int foo (int num1, int num2, int num3, int num4)
{
    int result;
    result = (num1 + num2) - (num3 + num4);
    return result;
}

void main(void)
{
    array2 [0] = foo(val1, val2, val3, val4[1]);
}

```

```

.globl val4
.data
.align 2
.type val4, @object
.size val4, 12

val4:
.word 7
.word 8
.word 9

.globl string1
.section .sdata
.align 2
.type string1, @object
.size string1, 4

string1:
.string "abc"

.globl string2
.align 2
.type string2, @object
.size string2, 3

string2:
.byte 97
.byte 98
.byte 99

```

23



SECURE, TRUSTED, AND ASSURED MICROELECTRONICS



Arizona State University

RISC-V Assembly – Let's Go All In

```

int val1 = 5;
int val2;
int val3 = 2;
int val4[3] = {7,8,9};

char string1[] = "abc";
char string2[3] = {'a','b','c'};

char array1[5];
int array2[2];
short int array3[3];

int foo (int num1, int num2, int num3, int num4)
{
    int result;
    result = (num1 + num2) - (num3 + num4);
    return result;
}

void main(void)
{
    array2 [0] = foo(val1, val2, val3, val4[1]);
}

```

```

.globl val4
.data
.align 2
.type val4, @object
.size val4, 12

val4:
.word 7
.word 8
.word 9

.globl string1
.section .sdata
.align 2
.type string1, @object
.size string1, 4

string1:
.string "abc"

.globl string2
.align 2
.type string2, @object
.size string2, 3

string2:
.byte 97
.byte 98
.byte 99

```

24



SECURE, TRUSTED, AND ASSURED MICROELECTRONICS



Arizona State University

RISC-V Assembly – Let's Go All In

```

int val1 = 5;
int val2 = 2;
int val3 = 7;
int val4[3] = {7,8,9};

char string[] = "abc";
char string2[3] = {'a','b','c'};

char array1[5];
int array2[2];
short int array3[3];

int foo (int num1, int num2, int num3, int num4)
{
    int result;
    result = (num1 + num2) - (num3 + num4);
    return result;
}

void main(void)
{
    array2[0] = foo(val1, val2, val3, val4[1]);
}

```

```

.section .array1, .sbss
.align 2
.type array1, 80000000
.size array1, 5
array1:
.zero 5

.section .array2, .sbss
.align 2
.type array2, 80000000
.size array2, 8
array2:
.zero 8

.section .array3, .sbss
.align 2
.type array3, 80000000
.size array3, 6
array3:
.zero 6

```

25



SECURE, TRUSTED, AND ASSURED MICROELECTRONICS



Arizona State University

RISC-V Assembly – Let's Go All In

```

int val1 = 5;
int val2 = 2;
int val3 = 7;
int val4[3] = {7,8,9};

char string[] = "abc";
char string2[3] = {'a','b','c'};

char array1[5];
int array2[2];
short int array3[3];

int foo (int num1, int num2, int num3, int num4)
{
    int result;
    result = (num1 + num2) - (num3 + num4);
    return result;
}

void main(void)
{
    array2[0] = foo(val1, val2, val3, val4[1]);
}

```

```

.text
.align 1
.global foo
.type foo, 8function
foo:
    addi sp,sp,-48
    sw s0,44(sp)
    addi s0,s0,48
    sw s0,-36(s0)
    sw a1,-40(s0)
    sw a2,-44(s0)
    sw a3,-48(s0)
    sw a4,-52(s0)
    lw a5,-40(s0)
    add a4,a4,a5
    lw a3,-44(s0)
    lw a2,-48(s0)
    add a3,a3,a5
    sub a5,a4,a5
    sw a5,-20(s0)
    lw a5,-20(s0)
    mv s0,a5
    lw s0,44(sp)
    addi sp,sp,48
    jr ra

```

26



SECURE, TRUSTED, AND ASSURED MICROELECTRONICS



Arizona State University

RISC-V Assembly – Let's Go All In

```

int val1 = 5;
int val2 = 2;
int val3 = 7;
int val4[3] = {7,8,9};

char string[] = "abc";
char string2[3] = {'a','b','c'};

char array1[5];
int array2[2];
short int array3[3];

int foo (int num1, int num2, int num3, int num4)
{
    int result;
    result = (num1 + num2) - (num3 + num4);
    return result;
}

void main(void)
{
    array2[0] = foo(val1, val2, val3, val4[1]);
}

```

```

.text
.align 1
.global foo
.type foo, 8function
foo:
    addi sp,sp,-48
    sw s0,44(sp)
    addi s0,s0,48
    sw s0,-36(s0)
    sw a1,-40(s0)
    sw a2,-44(s0)
    sw a3,-48(s0)
    sw a4,-52(s0)
    lw a5,-40(s0)
    add a4,a4,a5
    lw a3,-44(s0)
    lw a2,-48(s0)
    add a3,a3,a5
    sub a5,a4,a5
    sw a5,-20(s0)
    lw a5,-20(s0)
    mv s0,a5
    lw s0,44(sp)
    addi sp,sp,48
    jr ra

```

Stack pointer adjustment

27

STAM Center SECURE, TRUSTED, AND ASSURED MICROELECTRONICS **ASU Engineering** Arizona State University

RISC-V Assembly – Let's Go All In

```

int val1 = 5;
int val2;
int val3 = 2;
int val4[3] = {7,8,9};

char string[] = "abc";
char string[3] = {'a','b'};

char array[15];
int array[2];
short int array[3];

int foo (int num1, int num2, int num3, int num4)
{
    int result;
    result = (num1 + num2) - (num3 + num4);
    return result;
}

void main(void)
{
    array2[0] = foo(val1, val2, val3, val4[1]);
}

```

```

.text
.align 2
.globl foo, @function
.type foo, @function
foo:
    addi sp,sp,-48
    addi sp,sp,48
    sw a0,-36(s0)
    sw a1,-40(s0)
    sw a2,-44(s0)
    sw a3,-48(s0)
    lw a5,-40(s0)
    add a4,a4,a5
    lw a3,-44(s0)
    lw a5,-48(s0)
    add a3,a3,a5
    sub a5,a4,a5
    sw a5,-20(s0)
    lw a5,-20(s0)
    mv a0,a5
    addi sp,sp,48
    jr ra

```

- Save old frame pointer
 - Caller's frame pointer

28

STAM Center SECURE, TRUSTED, AND ASSURED MICROELECTRONICS **ASU Engineering** Arizona State University

RISC-V Assembly – Let's Go All In

```

int val1 = 5;
int val2;
int val3 = 2;
int val4[3] = {7,8,9};

char string[] = "abc";
char string[3] = {'a','b'};

char array[15];
int array[2];
short int array[3];

int foo (int num1, int num2, int num3, int num4)
{
    int result;
    result = (num1 + num2) - (num3 + num4);
    return result;
}

void main(void)
{
    array2[0] = foo(val1, val2, val3, val4[1]);
}

```

```

.text
.align 2
.globl foo, @function
.type foo, @function
foo:
    addi sp,sp,-48
    addi sp,sp,48
    sw a0,-36(s0)
    sw a1,-40(s0)
    sw a2,-44(s0)
    sw a3,-48(s0)
    lw a5,-40(s0)
    add a4,a4,a5
    lw a3,-44(s0)
    lw a5,-48(s0)
    add a3,a3,a5
    sub a5,a4,a5
    sw a5,-20(s0)
    lw a5,-20(s0)
    mv a0,a5
    addi sp,sp,48
    jr ra

```

- The frame pointer (fp/s0) points to the start of the stack frame
 - It points to the base of the stack frame
 - Parameters passed in to the subroutine remain at constant places relative to the frame pointer
 - It does not move for the duration of the subroutine call

29

STAM Center SECURE, TRUSTED, AND ASSURED MICROELECTRONICS **ASU Engineering** Arizona State University

RISC-V Assembly – Let's Go All In

```

int val1 = 5;
int val2;
int val3 = 2;
int val4[3] = {7,8,9};

char string[] = "abc";
char string[3] = {'a','b'};

char array[15];
int array[2];
short int array[3];

int foo (int num1, int num2, int num3, int num4)
{
    int result;
    result = (num1 + num2) - (num3 + num4);
    return result;
}

void main(void)
{
    array2[0] = foo(val1, val2, val3, val4[1]);
}

```

```

.text
.align 2
.globl foo, @function
.type foo, @function
foo:
    addi sp,sp,-48
    addi sp,sp,48
    sw a0,-36(s0)
    sw a1,-40(s0)
    sw a2,-44(s0)
    sw a3,-48(s0)
    lw a5,-40(s0)
    add a4,a4,a5
    lw a3,-44(s0)
    lw a5,-48(s0)
    add a3,a3,a5
    sub a5,a4,a5
    sw a5,-20(s0)
    lw a5,-20(s0)
    mv a0,a5
    addi sp,sp,48
    jr ra

```

- Save the function arguments
 - In case, something happens in the middle of the function execution
 - Like divide by zero

30

STAM Center

SECURE, TRUSTED, AND ASSURED MICROELECTRONICS

ASU Engineering

Arizona State University

RISC-V Assembly – Let's Go All In

```

int val1 = 5;
int val2;
int val3 = 2;
int val4[3] = {7,8,9};

char string1[] = "abc";
char string2[3] = {'a','b','c'};

char array1[5];
int array2[2];
short int array3[3];

int foo (int num1, int num2, int num3, int num4)
{
    int result;
    result = (num1 + num2) - (num3 + num4);
    return result;
}

void main(void) {
    array2 [0] = foo(val1, val2, val3, val4[1]);
}

```

```

.text
.globl foo, @function
.type foo, @function
foo:
    addi    sp,sp,-48
    sw      a0,44(sp)
    addi    a0,sp,48
    sw      a1,-36(s0)
    sw      a2,-44(s0)
    sw      a3,-48(s0)
    lw      a4,-36(s0)
    add     a5,-40(s0)
    add     a5,a4,a5
    sub     a5,a4,a5
    lw      a3,-20(s0)
    lw      a3,-44(s0)
    sw      a0,44(sp)
    addi    sp,sp,48
    jr      ra

```

Perform actual operations

5

31

STAM Center

SECURE, TRUSTED, AND ASSURED MICROELECTRONICS

ASU Engineering

Arizona State University

RISC-V Assembly – Let's Go All In

```

int val1 = 5;
int val2;
int val3 = 2;
int val4[3] = {7,8,9};

char string1[] = "abc";
char string2[3] = {'a','b','c'};

char array1[5];
int array2[2];
short int array3[3];

int foo (int num1, int num2, int num3, int num4)
{
    int result;
    result = (num1 + num2) - (num3 + num4);
    return result;
}

void main(void) {
    array2 [0] = foo(val1, val2, val3, val4[1]);
}

```

```

.text
.globl foo, @function
.type foo, @function
foo:
    addi    sp,sp,-48
    sw      a0,44(sp)
    addi    a0,sp,48
    sw      a1,-36(s0)
    sw      a2,-40(s0)
    sw      a2,-44(s0)
    sw      a3,-48(s0)
    lw      a4,-36(s0)
    add     a5,-40(s0)
    add     a5,a4,a5
    sub     a5,a4,a5
    lw      a3,-20(s0)
    lw      a3,-44(s0)
    sw      a0,44(sp)
    addi    sp,sp,48
    jr      ra

```

Return result

6

32

STAM Center

SECURE, TRUSTED, AND ASSURED MICROELECTRONICS

ASU Engineering

Arizona State University

RISC-V Assembly – Let's Go All In

```

int val1 = 5;
int val2;
int val3 = 2;
int val4[3] = {7,8,9};

char string1[] = "abc";
char string2[3] = {'a','b','c'};

char array1[5];
int array2[2];
short int array3[3];

int foo (int num1, int num2, int num3, int num4)
{
    int result;
    result = (num1 + num2) - (num3 + num4);
    return result;
}

void main(void) {
    array2 [0] = foo(val1,

```

```

.text
.globl foo, @function
.type foo, @function
foo:
    addi    sp,sp,-48
    sw      a0,44(sp)
    addi    a0,sp,48
    sw      a1,-36(s0)
    sw      a2,-40(s0)
    sw      a2,-44(s0)
    sw      a3,-48(s0)
    lw      a4,-36(s0)
    add     a5,-40(s0)
    add     a5,a4,a5
    sub     a5,a4,a5
    lw      a3,-20(s0)
    lw      a3,-44(s0)
    sw      a0,44(sp)
    addi    sp,sp,48
    jr      ra

```

Rollback things (frame pointer and stack pointer)

7

33

Stack Segment
(grows down, decreasing addresses)
Contains: local variables, function arguments, return addresses, and local variables.

Free space

Heap Segment
(grows up, increasing addresses)
Contains: dynamically allocated memory (malloc, free).

BSS Segment
(uninitialized data)
Contains: uninitialized global and static variables.

Data Segment
(initialized data)
Contains: initialized global and static variables.

Text Segment
(code)
Contains: machine code.

RISC-V Assembly – Let's Go All In

```

int val1 = 5;
int val2;
int val3 = 2;
int val4[3] = {7,8,9};

char string[] = "abc";
char string2[] = {'a','b','c'};

char array[15];
int array2[2];
short int array3[3];

int foo (int num1, int num2, int num3, int num4)
{
    int result;
    result = (num1 + num2) - (num3 + num4);
    return result;
}

void main(void) {
    array2[0] = foo(val1, val2, val3, val4[1]);
}
    
```

The frame pointer (`fp`/`$0`) points to the start of the stack frame.

- It points to the base of the stack frame.
- It is restored at the end of the call to its previous value.

```

.section .text
.globl foo, @function
.type foo, @function

foo:
    addi sp,sp,-48
    sw $0,44(sp)
    addi $0,$0,48

    # Prologue
    sw $1,-40($0)
    sw $2,-36($0)
    sw $3,-32($0)
    sw $4,-28($0)
    lw $5,-24($0)
    add $4,$4,$5
    lw $3,-20($0)
    lw $2,-16($0)
    add $3,$3,$2
    sub $5,$4,$5
    sw $5,-20($0)
    lw $5,-20($0)
    mv $0,$5
    addi $0,$0,48
    jr ra

    # Epilogue
    
```

34

Stack Segment
(grows down, decreasing addresses)
Contains: local variables, function arguments, return addresses, and local variables.

Free space

Heap Segment
(grows up, increasing addresses)
Contains: dynamically allocated memory (malloc, free).

BSS Segment
(uninitialized data)
Contains: uninitialized global and static variables.

Data Segment
(initialized data)
Contains: initialized global and static variables.

Text Segment
(code)
Contains: machine code.

RISC-V Assembly – Let's Go All In

```

int val1 = 5;
int val2;
int val3 = 2;
int val4[3] = {7,8,9};

char string[] = "abc";
char string2[] = {'a','b','c'};

char array[15];
int array2[2];
short int array3[3];

int foo (int num1, int num2, int num3, int num4)
{
    int result;
    result = (num1 + num2) - (num3 + num4);
    return result;
}

void main(void) {
    array2[0] = foo(val1, val2, val3, val4[1]);
}
    
```

Jump out and time to go Albert!

```

.section .text
.globl foo, @function
.type foo, @function

foo:
    addi sp,sp,-48
    sw $0,44(sp)
    addi $0,$0,48

    # Prologue
    sw $1,-40($0)
    sw $2,-36($0)
    sw $3,-32($0)
    sw $4,-28($0)
    lw $5,-24($0)
    add $4,$4,$5
    lw $3,-20($0)
    lw $2,-16($0)
    add $3,$3,$2
    sub $5,$4,$5
    sw $5,-20($0)
    lw $5,-20($0)
    mv $0,$5
    addi $0,$0,48
    jr ra

    # Epilogue
    
```

35

Stack Segment
(grows down, decreasing addresses)
Contains: local variables, function arguments, return addresses, and local variables.

Free space

Heap Segment
(grows up, increasing addresses)
Contains: dynamically allocated memory (malloc, free).

BSS Segment
(uninitialized data)
Contains: uninitialized global and static variables.

Data Segment
(initialized data)
Contains: initialized global and static variables.

Text Segment
(code)
Contains: machine code.

RISC-V Assembly – Let's Go All In

```

int val1 = 5;
int val2;
int val3 = 2;
int val4[3] = {7,8,9};

char string[] = "abc";
char string2[] = {'a','b','c'};

char array[15];
int array2[2];
short int array3[3];

int foo (int num1, int num2, int num3, int num4)
{
    int result;
    result = (num1 + num2) - (num3 + num4);
    return result;
}

void main(void) {
    array2[0] = foo(val1, val2, val3, val4[1]);
}
    
```

Jump out and time to go Albert!

```

.section .text
.globl foo, @function
.type foo, @function

foo:
    addi sp,sp,-48
    sw $0,44(sp)
    addi $0,$0,48

    # Prologue
    sw $1,-40($0)
    sw $2,-36($0)
    sw $3,-32($0)
    sw $4,-28($0)
    lw $5,-24($0)
    add $4,$4,$5
    lw $3,-20($0)
    lw $2,-16($0)
    add $3,$3,$2
    sub $5,$4,$5
    sw $5,-20($0)
    lw $5,-20($0)
    mv $0,$5
    addi $0,$0,48
    jr ra

    # Epilogue
    
```

36



SECURE, TRUSTED, AND ASSURED MICROELECTRONICS



Arizona State University

RISC-V Assembly – Let's Go All In

```

int val1 = 5;
int val2 = 2;
int val3 = 7;
int val4[3] = {7,8,9};

char string[] = "abc";
char string2[] = {'a','b','c'};

char array1[5];
int array2[2];
short int array3[3];

int foo(int num1, int num2, int num3, int num4)
{
    int result;
    result = (num1 + num2) - (num3 + num4);
    return result;
}

void main(void)
{
    array2[0] = foo(val1, val2, val3, val4[1]);
}

```

```

        .text
        .globl main
        .type main, @function

main:
    addi    sp,sp,-16
    mv      a0,a0
    mv      a1,a1
    mv      a2,a2
    mv      a3,a3
    mv      a4,a4
    mv      a5,a5
    mv      a6,a6
    mv      a7,a7
    mv      a8,a8
    mv      a9,a9
    mv      a10,a10
    mv      a11,a11
    mv      a12,a12
    mv      a13,a13
    mv      a14,a14
    mv      a15,a15
    mv      a16,a16
    mv      a17,a17
    mv      a18,a18
    mv      a19,a19
    mv      a20,a20
    mv      a21,a21
    mv      a22,a22
    mv      a23,a23
    mv      a24,a24
    mv      a25,a25
    mv      a26,a26
    mv      a27,a27
    mv      a28,a28
    mv      a29,a29
    mv      a30,a30
    mv      a31,a31
    mv      a32,a32
    mv      a33,a33
    mv      a34,a34
    mv      a35,a35
    mv      a36,a36
    mv      a37,a37
    mv      a38,a38
    mv      a39,a39
    mv      a40,a40
    mv      a41,a41
    mv      a42,a42
    mv      a43,a43
    mv      a44,a44
    mv      a45,a45
    mv      a46,a46
    mv      a47,a47
    mv      a48,a48
    mv      a49,a49
    mv      a50,a50
    mv      a51,a51
    mv      a52,a52
    mv      a53,a53
    mv      a54,a54
    mv      a55,a55
    mv      a56,a56
    mv      a57,a57
    mv      a58,a58
    mv      a59,a59
    mv      a60,a60
    mv      a61,a61
    mv      a62,a62
    mv      a63,a63
    mv      a64,a64
    mv      a65,a65
    mv      a66,a66
    mv      a67,a67
    mv      a68,a68
    mv      a69,a69
    mv      a70,a70
    mv      a71,a71
    mv      a72,a72
    mv      a73,a73
    mv      a74,a74
    mv      a75,a75
    mv      a76,a76
    mv      a77,a77
    mv      a78,a78
    mv      a79,a79
    mv      a80,a80
    mv      a81,a81
    mv      a82,a82
    mv      a83,a83
    mv      a84,a84
    mv      a85,a85
    mv      a86,a86
    mv      a87,a87
    mv      a88,a88
    mv      a89,a89
    mv      a90,a90
    mv      a91,a91
    mv      a92,a92
    mv      a93,a93
    mv      a94,a94
    mv      a95,a95
    mv      a96,a96
    mv      a97,a97
    mv      a98,a98
    mv      a99,a99
    mv      a100,a100
    mv      a101,a101
    mv      a102,a102
    mv      a103,a103
    mv      a104,a104
    mv      a105,a105
    mv      a106,a106
    mv      a107,a107
    mv      a108,a108
    mv      a109,a109
    mv      a110,a110
    mv      a111,a111
    mv      a112,a112
    mv      a113,a113
    mv      a114,a114
    mv      a115,a115
    mv      a116,a116
    mv      a117,a117
    mv      a118,a118
    mv      a119,a119
    mv      a120,a120
    mv      a121,a121
    mv      a122,a122
    mv      a123,a123
    mv      a124,a124
    mv      a125,a125
    mv      a126,a126
    mv      a127,a127
    mv      a128,a128
    mv      a129,a129
    mv      a130,a130
    mv      a131,a131
    mv      a132,a132
    mv      a133,a133
    mv      a134,a134
    mv      a135,a135
    mv      a136,a136
    mv      a137,a137
    mv      a138,a138
    mv      a139,a139
    mv      a140,a140
    mv      a141,a141
    mv      a142,a142
    mv      a143,a143
    mv      a144,a144
    mv      a145,a145
    mv      a146,a146
    mv      a147,a147
    mv      a148,a148
    mv      a149,a149
    mv      a150,a150
    mv      a151,a151
    mv      a152,a152
    mv      a153,a153
    mv      a154,a154
    mv      a155,a155
    mv      a156,a156
    mv      a157,a157
    mv      a158,a158
    mv      a159,a159
    mv      a160,a160
    mv      a161,a161
    mv      a162,a162
    mv      a163,a163
    mv      a164,a164
    mv      a165,a165
    mv      a166,a166
    mv      a167,a167
    mv      a168,a168
    mv      a169,a169
    mv      a170,a170
    mv      a171,a171
    mv      a172,a172
    mv      a173,a173
    mv      a174,a174
    mv      a175,a175
    mv      a176,a176
    mv      a177,a177
    mv      a178,a178
    mv      a179,a179
    mv      a180,a180
    mv      a181,a181
    mv      a182,a182
    mv      a183,a183
    mv      a184,a184
    mv      a185,a185
    mv      a186,a186
    mv      a187,a187
    mv      a188,a188
    mv      a189,a189
    mv      a190,a190
    mv      a191,a191
    mv      a192,a192
    mv      a193,a193
    mv      a194,a194
    mv      a195,a195
    mv      a196,a196
    mv      a197,a197
    mv      a198,a198
    mv      a199,a199
    mv      a200,a200
    mv      a201,a201
    mv      a202,a202
    mv      a203,a203
    mv      a204,a204
    mv      a205,a205
    mv      a206,a206
    mv      a207,a207
    mv      a208,a208
    mv      a209,a209
    mv      a210,a210
    mv      a211,a211
    mv      a212,a212
    mv      a213,a213
    mv      a214,a214
    mv      a215,a215
    mv      a216,a216
    mv      a217,a217
    mv      a218,a218
    mv      a219,a219
    mv      a220,a220
    mv      a221,a221
    mv      a222,a222
    mv      a223,a223
    mv      a224,a224
    mv      a225,a225
    mv      a226,a226
    mv      a227,a227
    mv      a228,a228
    mv      a229,a229
    mv      a230,a230
    mv      a231,a231
    mv      a232,a232
    mv      a233,a233
    mv      a234,a234
    mv      a235,a235
    mv      a236,a236
    mv      a237,a237
    mv      a238,a238
    mv      a239,a239
    mv      a240,a240
    mv      a241,a241
    mv      a242,a242
    mv      a243,a243
    mv      a244,a244
    mv      a245,a245
    mv      a246,a246
    mv      a247,a247
    mv      a248,a248
    mv      a249,a249
    mv      a250,a250
    mv      a251,a251
    mv      a252,a252
    mv      a253,a253
    mv      a254,a254
    mv      a255,a255
    mv      a256,a256
    mv      a257,a257
    mv      a258,a258
    mv      a259,a259
    mv      a260,a260
    mv      a261,a261
    mv      a262,a262
    mv      a263,a263
    mv      a264,a264
    mv      a265,a265
    mv      a266,a266
    mv      a267,a267
    mv      a268,a268
    mv      a269,a269
    mv      a270,a270
    mv      a271,a271
    mv      a272,a272
    mv      a273,a273
    mv      a274,a274
    mv      a275,a275
    mv      a276,a276
    mv      a277,a277
    mv      a278,a278
    mv      a279,a279
    mv      a280,a280
    mv      a281,a281
    mv      a282,a282
    mv      a283,a283
    mv      a284,a284
    mv      a285,a285
    mv      a286,a286
    mv      a287,a287
    mv      a288,a288
    mv      a289,a289
    mv      a290,a290
    mv      a291,a291
    mv      a292,a292
    mv      a293,a293
    mv      a294,a294
    mv      a295,a295
    mv      a296,a296
    mv      a297,a297
    mv      a298,a298
    mv      a299,a299
    mv      a300,a300
    mv      a301,a301
    mv      a302,a302
    mv      a303,a303
    mv      a304,a304
    mv      a305,a305
    mv      a306,a306
    mv      a307,a307
    mv      a308,a308
    mv      a309,a309
    mv      a310,a310
    mv      a311,a311
    mv      a312,a312
    mv      a313,a313
    mv      a314,a314
    mv      a315,a315
    mv      a316,a316
    mv      a317,a317
    mv      a318,a318
    mv      a319,a319
    mv      a320,a320
    mv      a321,a321
    mv      a322,a322
    mv      a323,a323
    mv      a324,a324
    mv      a325,a325
    mv      a326,a326
    mv      a327,a327
    mv      a328,a328
    mv      a329,a329
    mv      a330,a330
    mv      a331,a331
    mv      a332,a332
    mv      a333,a333
    mv      a334,a334
    mv      a335,a335
    mv      a336,a336
    mv      a337,a337
    mv      a338,a338
    mv      a339,a339
    mv      a340,a340
    mv      a341,a341
    mv      a342,a342
    mv      a343,a343
    mv      a344,a344
    mv      a345,a345
    mv      a346,a346
    mv      a347,a347
    mv      a348,a348
    mv      a349,a349
    mv      a350,a350
    mv      a351,a351
    mv      a352,a352
    mv      a353,a353
    mv      a354,a354
    mv      a355,a355
    mv      a356,a356
    mv      a357,a357
    mv      a358,a358
    mv      a359,a359
    mv      a360,a360
    mv      a361,a361
    mv      a362,a362
    mv      a363,a363
    mv      a364,a364
    mv      a365,a365
    mv      a366,a366
    mv      a367,a367
    mv      a368,a368
    mv      a369,a369
    mv      a370,a370
    mv      a371,a371
    mv      a372,a372
    mv      a373,a373
    mv      a374,a374
    mv      a375,a375
    mv      a376,a376
    mv      a377,a377
    mv      a378,a378
    mv      a379,a379
    mv      a380,a380
    mv      a381,a381
    mv      a382,a382
    mv      a383,a383
    mv      a384,a384
    mv      a385,a385
    mv      a386,a386
    mv      a387,a387
    mv      a388,a388
    mv      a389,a389
    mv      a390,a390
    mv      a391,a391
    mv      a392,a392
    mv      a393,a393
    mv      a394,a394
    mv      a395,a395
    mv      a396,a396
    mv      a397,a397
    mv      a398,a398
    mv      a399,a399
    mv      a400,a400
    mv      a401,a401
    mv      a402,a402
    mv      a403,a403
    mv      a404,a404
    mv      a405,a405
    mv      a406,a406
    mv      a407,a407
    mv      a408,a408
    mv      a409,a409
    mv      a410,a410
    mv      a411,a411
    mv      a412,a412
    mv      a413,a413
    mv      a414,a414
    mv      a415,a415
    mv      a416,a416
    mv      a417,a417
    mv      a418,a418
    mv      a419,a419
    mv      a420,a420
    mv      a421,a421
    mv      a422,a422
    mv      a423,a423
    mv      a424,a424
    mv      a425,a425
    mv      a426,a426
    mv      a427,a427
    mv      a428,a428
    mv      a429,a429
    mv      a430,a430
    mv      a431,a431
    mv      a432,a432
    mv      a433,a433
    mv      a434,a434
    mv      a435,a435
    mv      a436,a436
    mv      a437,a437
    mv      a438,a438
    mv      a439,a439
    mv      a440,a440
    mv      a441,a441
    mv      a442,a442
    mv      a443,a443
    mv      a444,a444
    mv      a445,a445
    mv      a446,a446
    mv      a447,a447
    mv      a448,a448
    mv      a449,a449
    mv      a450,a450
    mv      a451,a451
    mv      a452,a452
    mv      a453,a453
    mv      a454,a454
    mv      a455,a455
    mv      a456,a456
    mv      a457,a457
    mv      a458,a458
    mv      a459,a459
    mv      a460,a460
    mv      a461,a461
    mv      a462,a462
    mv      a463,a463
    mv      a464,a464
    mv      a465,a465
    mv      a466,a466
    mv      a467,a467
    mv      a468,a468
    mv      a469,a469
    mv      a470,a470
    mv      a471,a471
    mv      a472,a472
    mv      a473,a473
    mv      a474,a474
    mv      a475,a475
    mv      a476,a476
    mv      a477,a477
    mv      a478,a478
    mv      a479,a479
    mv      a480,a480
    mv      a481,a481
    mv      a482,a482
    mv      a483,a483
    mv      a484,a484
    mv      a485,a485
    mv      a486,a486
    mv      a487,a487
    mv      a488,a488
    mv      a489,a489
    mv      a490,a490
    mv      a491,a491
    mv      a492,a492
    mv      a493,a493
    mv      a494,a494
    mv      a495,a495
    mv      a496,a496
    mv      a497,a497
    mv      a498,a498
    mv      a499,a499
    mv      a500,a500
    mv      a501,a501
    mv      a502,a502
    mv      a503,a503
    mv      a504,a504
    mv      a505,a505
    mv      a506,a506
    mv      a507,a507
    mv      a508,a508
    mv      a509,a509
    mv      a510,a510
    mv      a511,a511
    mv      a512,a512
    mv      a513,a513
    mv      a514,a514
    mv      a515,a515
    mv      a516,a516
    mv      a517,a517
    mv      a518,a518
    mv      a519,a519
    mv      a520,a520
    mv      a521,a521
    mv      a522,a522
    mv      a523,a523
    mv      a524,a524
    mv      a525,a525
    mv      a526,a526
    mv      a527,a527
    mv      a528,a528
    mv      a529,a529
    mv      a530,a530
    mv      a531,a531
    mv      a532,a532
    mv      a533,a533
    mv      a534,a534
    mv      a535,a535
    mv      a536,a536
    mv      a537,a537
    mv      a538,a538
    mv      a539,a539
    mv      a540,a540
    mv      a541,a541
    mv      a542,a542
    mv      a543,a543
    mv      a544,a544
    mv      a545,a545
    mv      a546,a546
    mv      a547,a547
    mv      a548,a548
    mv      a549,a549
    mv      a550,a550
    mv      a551,a551
    mv      a552,a552
    mv      a553,a553
    mv      a554,a554
    mv      a555,a555
    mv      a556,a556
    mv      a557,a557
    mv      a558,a558
    mv      a559,a559
    mv      a560,a560
    mv      a561,a561
    mv      a562,a562
    mv      a563,a563
    mv      a564,a564
    mv      a565,a565
    mv      a566,a566
    mv      a567,a567
    mv      a568,a568
    mv      a569,a569
    mv      a570,a570
    mv      a571,a571
    mv      a572,a572
    mv      a573,a573
    mv      a574,a574
    mv      a575,a575
    mv      a576,a576
    mv      a577,a577
    mv      a578,a578
    mv      a579,a579
    mv      a580,a580
    mv      a581,a581
    mv      a582,a582
    mv      a583,a583
    mv      a584,a584
    mv      a585,a585
    mv      a586,a586
    mv      a587,a587
    mv      a588,a588
    mv      a589,a589
    mv      a590,a590
    mv      a591,a591
    mv      a592,a592
    mv      a593,a593
    mv      a594,a594
    mv      a595,a595
    mv      a596,a596
    mv      a597,a597
    mv      a598,a598
    mv      a599,a599
    mv      a600,a600
    mv      a601,a601
    mv      a602,a602
    mv      a603,a603
    mv      a604,a604
    mv      a605,a605
    mv      a606,a606
    mv      a607,a607
    mv      a608,a608
    mv      a609,a609
    mv      a610,a610
    mv      a611,a611
    mv      a612,a612
    mv      a613,a613
    mv      a614,a614
    mv      a615,a615
    mv      a616,a616
    mv      a617,a617
    mv      a618,a618
    mv      a619,a619
    mv      a620,a620
    mv      a621,a621
    mv      a622,a622
    mv      a623,a623
    mv      a624,a624
    mv      a625,a625
    mv      a626,a626
    mv      a627,a627
    mv      a628,a628
    mv      a629,a629
    mv      a630,a630
    mv      a631,a631
    mv      a632,a632
    mv      a633,a633
    mv      a634,a634
    mv      a635,a635
    mv      a636,a636
    mv      a637,a637
    mv      a638,a638
    mv      a639,a639
    mv      a640,a640
    mv      a641,a641
    mv      a642,a642
    mv      a643,a643
    mv      a644,a644
    mv      a645,a645
    mv      a646,a646
    mv      a647,a647
    mv      a648,a648
    mv      a649,a649
    mv      a650,a650
    mv      a651,a651
    mv      a652,a652
    mv      a653,a653
    mv      a654,a654
    mv      a655,a655
    mv      a656,a656
    mv      a657,a657
    mv      a658,a658
    mv      a659,a659
    mv      a660,a660
    mv      a661,a661
    mv      a662,a662
    mv      a663,a663
    mv      a664,a664
    mv      a665,a665
    mv      a666,a666
    mv      a667,a667
    mv      a668,a668
    mv      a669,a669
    mv      a670,a670
    mv      a671,a671
    mv      a672,a672
    mv      a673,a673
    mv      a674,a674
    mv      a675,a675
    mv      a676,a676
    mv      a677,a677
    mv      a678,a678
    mv      a679,a679
    mv      a680,a680
    mv      a681,a681
    mv      a682,a682
    mv      a683,a683
    mv      a684,a684
    mv      a685,a685
    mv      a686,a686
    mv      a687,a687
    mv      a688,a688
    mv      a689,a689
    mv      a690,a690
    mv      a691,a691
    mv      a692,a692
    mv      a693,a693
    mv      a694,a694
    mv      a695,a695
    mv      a696,a696
    mv      a697,a697
    mv      a698,a698
    mv      a699,a699
    mv      a700,a700
    mv      a701,a701
    mv      a702,a702
    mv      a703,a703
    mv      a704,a704
    mv      a705,a705
    mv      a706,a706
    mv      a707,a707
    mv      a708,a708
    mv      a709,a709
    mv      a710,a710
    mv      a711,a711
    mv      a712,a712
    mv      a713,a713
    mv      a714,a714
    mv      a715,a715
    mv      a716,a716
    mv      a717,a717
    mv      a718,a718
    mv      a719,a719
    mv      a720,a720
    mv      a721,a721
    mv      a722,a722
    mv      a723,a723
    mv      a724,a724
    mv      a725,a725
    mv      a726,a726
    mv      a727,a727
    mv      a728,a728
    mv      a729,a729
    mv      a730,a730
    mv      a731,a731
    mv      a732,a732
    mv      a733,a733
    mv      a734,a734
    mv      a735,a735
    mv      a736,a736
    mv      a737,a737
    mv      a738,a738
    mv      a739,a739
    mv      a740,a740
    mv      a741,a741
    mv      a742,a742
    mv      a743,a743
    mv      a744,a744
    mv      a745,a745
    mv      a746,a746
    mv      a747,a747
    mv      a748,a748
    mv      a749,a749
    mv      a750,a750
    mv      a751,a751
    mv      a752,a752
    mv      a753,a753
    mv      a754,a754
    mv      a755,a755
    mv      a756,a756
    mv      a757,a757
    mv      a758,a758
    mv      a759,a759
    mv      a760,a760
    mv      a761,a761
    mv      a762,a762
    mv      a763,a763
    mv      a764,a764
    mv      a765,a765
    mv      a766,a766
    mv      a767,a767
    mv      a768,a768
    mv      a769,a769
    mv      a770,a770
    mv      a771,a771
    mv      a772,a772
    mv      a773,a773
    mv      a774,a774
    mv      a775,a775
    mv      a776,a776
    mv      a777,a777
    mv      a778,a778
    mv      a779,a779
    mv      a780,a780
    mv      a781,a781
    mv      a782,a782
    mv      a783,a783
    mv      a784,a784
    mv      a785,a785
    mv      a786,a786
    mv      a787,a787
    mv      a788,a788
    mv      a789,a789
    mv      a790,a790
    mv      a791,a791
    mv      a792,a792
    mv      a793,a793
    mv      a794,a794
    mv      a795,a795
    mv      a796,a796
    mv      a797,a797
    mv      a798,a798
    mv      a799,a799
    mv      a800,a800
    mv      a801,a801
    mv      a802,a802
    mv      a803,a803
    mv      a804,a804
    mv      a805,a805
    mv      a806,a806
    mv      a807,a807
    mv      a808,a808
    mv      a809,a809
    mv      a810,a810
    mv      a811,a811
    mv      a812,a812
    mv      a813,a813
    mv      a814,a814
    mv      a815,a815
    mv      a816,a816
    mv      a817,a817
    mv      a818,a818
    mv      a819,a819
    mv      a820,a820
    mv      a821,a821
    mv      a822,a822
    mv      a823,a823
    mv      a824,a824
    mv      a825,a825
    mv      a826,a826
    mv      a827,a827
    mv      a828,a828
    mv      a829,a829
    mv      a830,a830
    mv      a831,a831
    mv      a832,a832
    mv      a833,a833
    mv      a834,a834
    mv      a835,a835
    mv      a836,a836
    mv      a837,a837
    mv      a838,a838
    mv      a839,a839
    mv      a840,a840
    mv      a841,a841
    mv      a842,a842
    mv      a843,a843
    mv      a844,a844
    mv      a845,a845
    mv      a846,a846
    mv      a847,a847
    mv      a848,a848
    mv      a849,a849
    mv      a850,a850
    mv      a851,a851
    mv      a852,a852
    mv      a853,a853
    mv      a854,a854
    mv      a855,a855
    mv      a856,a856
    mv      a857,a857
    mv      a858,a858
    mv      a859,a859
    mv      a860,a860
    mv      a861,a861
    mv      a862,a862
    mv      a863,a863
    mv      a864,a864
    mv      a865,a865
    mv      a866,a866
    mv      a867,a867
    mv      a868,a868
    mv      a869,a869
    mv      a870,a870
    mv      a871,a871
    mv      a872,a872
    mv      a873,a873
    mv      a874,a874
    mv      a875,a875
    mv      a876,a876
    mv      a877,a877
    mv      a878,a878
    mv      a879,a879
    mv      a880,a880
    mv      a881,a881
    mv      a882,a882
    mv      a883,a883
    mv      a884,a884
    mv      a885,a885
    mv      a886,a886
    mv      a887,a887
    mv      a888,a888
    mv      a889,a889
    mv      a890,a890
    mv      a891,a891
    mv      a892,a892
    mv      a893,a893
    mv      a894,a894
    mv      a895,a895
    mv      a896,a896
    mv      a897,a897
    mv      a898,a898
    mv      a899,a899
    mv      a900,a900
    mv      a901,a901
    mv      a902,a902
    mv      a903,a903
    mv      a904,a904
    mv      a905,a905
    mv      a906,a906
    mv      a907,a907
    mv      a908,a908
    mv      a909,a909
    mv      a910,a910
    mv      a911,a911
    mv      a912,a912
    mv      a913,a913
    mv      a914,a914
    mv      a915,a915
    mv      a916,a916
    mv      a917,a917
    mv      a918,a918
    mv      a919,a919
    mv      a920,a920
    mv      a921,a921
    mv      a922,a922
    mv      a923,a923
    mv      a924,a924
    mv      a925,a925
    mv      a926,a926
    mv      a927,a927
    mv      a928,a928
    mv      a929,a929
    mv      a930,a930
    mv      a931,a931
    mv      a932,a932
    mv      a933,a933
    mv      a934,a934
    mv      a935,a935
    mv      a936,a936
    mv      a937,a937
    mv      a938,a938
    mv      a939,a939
    mv      a940,a940
    mv      a941,a941
    mv      a942,a942
    mv      a943,a943
    mv      a944,a944
    mv      a945,a945
    mv      a946,a946
    mv      a947,a947
    mv      a948,a948
    mv      a949,a949
    mv      a950,a950
    mv      a951,a951
    mv      a952,a952
    mv      a953,a953
    mv      a954,a954
    mv      a955,a955
    mv      a956,a956
    mv      a957,a957
    mv      a958,a958
    mv      a959,a959
    mv      a960,a960
    mv      a961,a961
    mv      a962,a962
    mv      a963,a963
    mv      a964,a964
    mv      a965,a965
    mv      a966,a966
    mv      a967,a967
    mv      a968,a968
    mv      a969,a969
    mv      a970,a970
    mv      a971,a971
    mv      a972,a972
    mv      a973,a973
    mv      a974,a974
    mv      a975,a975
    mv      a976,a976
    mv      a977,a977
    mv      a978,a978
    mv      a979,a979
    mv      a980,a980
    mv      a981,a981
    mv      a982,a982
    mv      a983,a983
    mv      a984,a984
    mv      a985,a985
    mv      a986,a986
    mv      a987,a987
    mv      a988,a988
    mv      a989,a989
    mv      a990,a990
    mv      a991,a991
    mv      a992,a992
    mv      a993,a993
    mv      a994,a994
    mv      a995,a995
    mv      a996,a996
    mv      a997,a997
    mv      a998,a998
    mv      a999,a999
    mv      a1000,a1000
    mv      a1001,a1001
    mv      a1002,a1002
    mv      a1003,a1003
    mv      a1004,a1004
    mv      a1005,a1005
    mv      a1006,a1006
    mv      a1007,a1007
    mv      a1008,a1008
    mv      a1009,a1009
    mv      a1010,a1010
    mv      a1011,a1011
    mv      a1012,a1012
    mv      a1013,a10
```



STAM Center
SECURE, TRUSTED, AND ASSURED MICROELECTRONICS



ASU Engineering
Arizona State University

Function Entry

```

prolog:
    addi    sp, sp, -16    # -16 (frame size) depends on the amount of space needed
    sw      ra, 12(sp)    # offset must be framesize-4
    sw      fp, 8(sp)     # store the caller's fp
    sw      gp, 4(sp)     # store caller's gp
    mv      fp, sp        # fp is now the new function's frame base pointer

# -
# The actual functional body of the function
# -

epilog:
    lw      gp, 4(sp)
    lw      fp, 8(sp)
    addi    sp, sp, 16    # reverse the effect of first instruction
    jr      ra
    
```

40



STAM Center
SECURE, TRUSTED, AND ASSURED MICROELECTRONICS



ASU Engineering
Arizona State University

Assembly Programs

- If (condition) then do this

```

if ( i == j )
    i++;
j++;
    
```

Assuming t1 stores i and t2 stores j:

```

bne t1, t2, NEXT # branch if !( i == j )
addi t1, t1, 1   # i++
NEXT: addi t2, t2, 1 # j++
        
```

41



STAM Center
SECURE, TRUSTED, AND ASSURED MICROELECTRONICS



ASU Engineering
Arizona State University

Assembly Programs

- If (condition) then do this else do that

```

if ( i == j && i == k )
    i++;
else
    j++;
j = i + k ;
    
```

Assuming t0 stores i, t1 stores j, and t2 stores k:

```

bne t0, t1, ELSE # cond1: branch if !( i == j )
bne t0, t2, ELSE # cond2: branch if !( i == k )
addi t0, t0, 1   # if-body: i++
j NEXT          # jump over else
ELSE: addi t1, t1, 1 # else-body: j++
NEXT: addi t1, t0, t2 # j = i + k
        
```

42

STAM Center
SECURE, TRUSTED, AND ASSURED MICROELECTRONICS

ASU Engineering
Arizona State University

Assembly Programs

- While (condition) then keep doing this

```

while ( i < j )
{
    k = k + 2;
    i++;
}
    
```

Assuming t0 stores i, t1 stores j, and t2 stores k:

```

Loop: bge t0, t1, DONE # branch if ! ( i < j )
      add t2, t2, 2     # k = k + 2
      addi t0, t0, 1    # i++
      j Loop           # jump back to top of loop
DONE:
    
```

43

STAM Center
SECURE, TRUSTED, AND ASSURED MICROELECTRONICS

ASU Engineering
Arizona State University

Assembly Programs

- For (these many times) do this

```

for ( i = 0; i < j; i++)
{
    k = k + 2;
}
    
```

Assuming t0 stores i, t1 stores j, and t2 stores k:

```

      addi t0, $0, 0    # i = 0
Loop: bge t0, t1, DONE # branch if ! ( i < j )
      add t2, t2, 2     # k = k + 2
      addi t0, t0, 1    # i++
      j Loop           # jump back to top of loop
DONE:
    
```

44

STAM Center
SECURE, TRUSTED, AND ASSURED MICROELECTRONICS

ASU Engineering
Arizona State University

Assembly Programs

- For (these many times) do this

```

switch( i ) {
    case 1:
        i++;
        break;
    case 2:
        i += 2;
        break;
    case 3:
        i += 3;
        break;
}
    
```

Assuming t0 stores i and t1 stores temp:

```

      addi t1, zero, 1 # case 1: set temp to 1
      bne t0, t1, CASE2 # false: branch to case 2
      j CASE1_BODY    # true: branch to case 1 body
CASE2:
      addi t1, zero, 2 # case 2: set temp to 2
      bne t0, t1, CASE3 # false: branch to case 3
      j CASE1_BODY    # true: branch to case 2 body
CASE3:
      addi t1, zero, 3 # case 3: set temp to 3
      bne t0, t1, EXIT # false: branch to exit
      j CASE1_BODY    # true: branch to case 3 body
CASE1_BODY:
      addi t0, t0, 1   # case 1 body: i++
      j EXIT          # break
CASE2_BODY:
      addi t0, t0, 2   # case 2 body: i += 2
      j EXIT          # break
CASE3_BODY:
      addi t0, t0, 3   # case 3 body: i += 3
      j EXIT          # break
EXIT:
    
```

45

STAM Center
SECURE, TRUSTED, AND ASSURED MICROELECTRONICS

ASU Engineering
Arizona State University

Assembly Programs

- Short program

```
clear(int array[], int size)
{
    int i;
    for (i = 0; i < size; i += 1)
        array[i] = 0;
}
```

```

mov t0,zero    # i = 0
Loop: sll t1,t0,2    # t1 = i * 4
      add t2,a0,t1    # t2 = &array[i]
      sw zero,0(t2)    # array[i] = 0
      addi t0,t0,1    # i = i + 1
      sll a3,t0,a1    # a3 = (i < size)
      bne a3,zero,Loop # if (...)
                        # goto loop

```

46

STAM Center
SECURE, TRUSTED, AND ASSURED MICROELECTRONICS

ASU Engineering
Arizona State University

Data Addressing Modes

- RISC-V has four data addressing modes
 - Register
 - Data is in a register
 - Immediate
 - Data is specified in the instruction directly
 - Displacement
 - Data is in memory with address calculated as Base + Offset
 - PC-Relative
 - Data is in memory with address calculated as PC + Offset

47

STAM Center
SECURE, TRUSTED, AND ASSURED MICROELECTRONICS

ASU Engineering
Arizona State University

Data Addressing Modes

- RISC-V has four data addressing modes
 - Memory related addressing modes
 - Register indirect
 - $x1 \leftarrow x2 + \text{Mem}[x1]$
 - Displacement
 - $x1 \leftarrow x2 + \text{Mem}[64 + x1]$

48

STAM Center
SECURE, TRUSTED, AND ASSURED MICROELECTRONICS

ASU Engineering
Arizona State University

Data Addressing Modes

- Other memory address modes (non RISC-V)
 - Direct or absolute**
 - add x1,(1024)
 - $x1 \leftarrow x1 + \text{Mem}[1024]$
 - Memory indirect**
 - add x1,@(x3)
 - $x1 \leftarrow x1 + \text{Mem}[\text{Mem}[x3]]$

Address	Data
3	100
2	17
1	134
0	2

49

STAM Center
SECURE, TRUSTED, AND ASSURED MICROELECTRONICS

ASU Engineering
Arizona State University

The Big Picture

```

#include <stdio.h>
int main()
{
    // Albert, this is a comment line!
    printf("Hello, Albert!\n");
    return 0;
}
    
```

C program → compiler → assembly code → assembler → object code → linker → executable → loader → memory

Library routines → linker → executable

Processor ↔ Memory (Data transfer)

50

STAM Center
SECURE, TRUSTED, AND ASSURED MICROELECTRONICS

ASU Engineering
Arizona State University

Application Compiling Process

- High-level language program (in C)


```

void swap (int array[], int i) {
    int temp;
    temp = array[i];
    array[i] = array[i+1];
    array[i+1] = temp;
}
            
```
- Assembly language program (for RISC-V)


```

swap:
    addi sp,sp,-48
    ...
    mv a5,a1
    ...
    ld s0,40(sp)
    addi sp,sp,48
    jr ra
            
```

one-to-many (arrow from C code to assembly code)

C compiler (circle around the transition)

51

Application Compiling Process

- Assembly language program (for RISC-V)

```

swap:
    addi sp, sp, -48
    ...
    mv a5, a1
    ...
    ld a0, 40(sp)
    addi sp, sp, 48
    jr ra
        
```
- Machine (object, binary) code (for RISC-V)

```

111111010000 00010 000 00010 0010011
000000110000 00010 000 01000 0010011
...
        
```

one-to-one

assembler

52

Application Compiling Process

- Assembler (or compiler) translates program into machine instructions
- Provides information for building a complete program from the pieces
 - Header: described contents of object module
 - Text segment: translated instructions
 - Static data segment: data allocated for the life of the program
 - Relocation info: for contents that depend on absolute location of loaded program
 - Symbol table: global definitions and external refs
 - Debug info: for associating with source code

53

Application Compiling Process

- Machine (object, binary) code (for RISC-V)

```

111111010000 00010 000 00010 0010011
000000110000 00010 000 01000 0010011
        
```
- Memory organization

Loader

54

Application Compiling Process

- Load from image file on disk into memory
 1. Read header to determine segment sizes
 2. Create virtual address space
 3. Copy text and initialized data into memory
 4. Set up arguments on stack
 5. Initialize registers (including sp, fp, gp)
 6. Jump to startup routine
- Copies arguments to a0, ... and calls main

55

The Big Idea in Today's Computers

- Stored Program Computer

Program = A sequence of instructions

 - Instructions represented in binary, just like data
 - Instructions and data stored in memory
 - Programs can operate on programs
 - e.g., compilers, linkers, ...
 - Binary compatibility allows compiled programs to work on different computers
 - Standardized ISAs

56

The Big Picture

```

#include <stdio.h>
int main()
{
    // Albert, this is a comment line!
    printf("hello, Albert!\n");
    return 0;
}

```

C program

↓ **compiler**

assembly code

↓ **assembler**

object code

↓ **linker**

executable

↓ **loader**

Let's start with the processor

Processor ↔ Data transfer ↔ Memory

Memory contains: Address, Data

Albert! It is time to build the computer from the ground up

57


SECURE, TRUSTED, AND ASSURED MICROELECTRONICS


Arizona State University

Next Lecture Module

- RISC-V ISA
