```
        .data
msg:    .asciiz "Hello World! \n "

        .text

## entry point of program must be called 'main'
main:  addiu $v0, $0, 4     # syscall 4 (print_str)
       la $a0 msg
       syscall
       addiu $v0, $0, 10
       syscall                    # syscall 10 (exit)
```

"Hello World" Program

'data' and 'text' areas

```
        .data
msg:    .asciiz "Hello World! \n "

        .text

## entry point of program must be called 'main'
main:  addiu $v0, $0, 4     # syscall 4 (print_str)
       la $a0 msg
       syscall
       addiu $v0, $0, 10
       syscall                    # syscall 10 (exit)
```

Syscalls -- used for I/O to console, exit.

$v0 holds syscall identification number

Argument registers used to pass values to syscall

'syscall' is like a software interrrupt

Syscalls

Service	Call Code	Arguments	Result
print_int	1	$a0=integer	
print_float	2	$f12=float	
print_double	3	$f12=double	
print_string	4	$a0=string	
read_int	5		int, (in $v0)
read_float	6		float (in $f0)
read_double	7		double (in $f0)
read_string	8	$a0=buffer, $a1=length	
sbrk	9	$a0 = amount	addr (in $v0)
exit	10		

la pseudoinstruction

la – ***load address***, is an assembler psuedoinstruction

Useful for placing the address of an operand into a register:

```
la   $a0   msg
```

Actually translates into:

```
lui   $1,  upper_16_bits_msg_addr
ori   $a0,  $1,  lower_16_bits_msg_addr
```

where value represents the starting address of the 'msg' operand. If lower half of address is '0000h', then the 'ori' instruction is not generated.

Note that Register '1' is used by the pseudoinstruction !!!!

Register Definitions

Name	Reg Number	Usage	Preserved on call?
$zero	0	constant value 0	N.A.
$v0-$v1	2-3	values for results, expression eval	no
$a0-$a3	4-7	arguments	yes
$t0-$t7	8-15	temporaries	no
$s0-$s7	16-23	saved	yes
$t8-$t9	24-25	more temps	no
$gp	28	global pointer	yes
$sp	29	stack pointer	yes
$fp	30	frame pointer	yes
$ra	31	rtn address	yes

Echo string program

```
        .data
msg1:   .asciiz "Enter a string: \n "
buff:   .space 64

      .text
## entry point of program must be called 'main'
main:  addiu $v0, $0, 4    # syscall 4 (print_str)
        la $a0 msg1            # prompt user
        syscall
       addiu $v0, $0, 4

        la  $a0 buff              #get string from user
        addiu $a1, $0, 64
        addiu $v0, $0, 8           #syscall 8 (read str)
       syscall

# echo string
        addiu $v0, $0, 4
        syscall
#exit

       addiu $v0, $0, 10
       syscall                    # syscall 10 (exit)
```

Implementing a Stack

SPIM initialization sets SP to high memory ($7FFFeffc)

To push a register value onto to the stack:

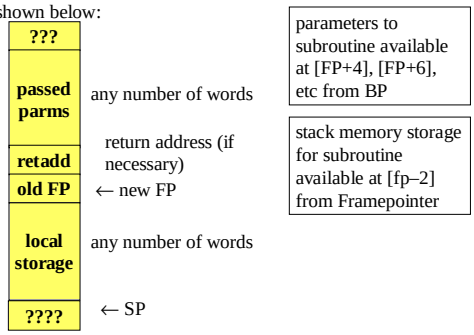
```
addiu  $sp,$sp,-4          # decrement stack pointer
sw     $6, 0($sp)          # push register 6 on stack
```

To pop a value off the stack

```
lw     $6, 0($sp)
addiu  $sp, $sp, 4         # increment stack pointer
```

A Stack Frame

A generalized picture of a *stack frame* used by a subroutine is shown below:



Code for Stack Frame (assume 2 words on stack as passed parameters)

```
suba:
      addiu $sp,$sp, -4
      sw   $ra, 0($sp)
      addiu $sp,$sp, -4
      sw   $fp, 0($sp)
      addiu $fp,$sp,$0
      addiu $sp, $sp,-8
      ...
      ...
      addiu $sp,$fp, 0
      lw   $fp, 0($sp)
      addiu $sp,$sp, 4
      lw   $ra, 0($sp)
      addiu $sp, $sp, 12
      $jr  $ra
```

Save RTN address (if needed)

Save old framepointer

Get new framepointer

Allocate local storage (if needed)

Restore SP, free local storage

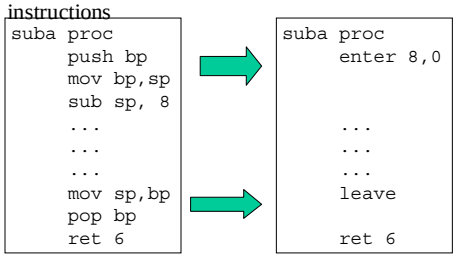
restore old framepointer

pop return address and clean stack of passed parameters

return

High Level Support for Stack Frames

- Some processors have special instructions that support stack frames
- x286 processors and higher has ‘Enter’ and ‘Leave’ instructions



Not the only possible arrangement of stack frames

- Optimizing compilers will take short cuts on stack frames
- Will leave out parts of frame if not needed
  - ⇒Saving of return address
  - ⇒Framepoint
  - ⇒Local Variable storage
  - ⇒Passing of arguments on stack
- One or more of these features can be left out