SNS COLLEGE OF TECHNOLOGY

(Autonomous)

COIMBATORE-35

Run time Storage Management

Run time Storage Management

The information which required during an execution of a procedure is kept in a block of storage called an activation record. The activation record includes storage for names local to the procedure.

We can describe address in the target code using the following ways:

Static allocation

Stack allocation

In static allocation, the position of an activation record is fixed in memory at compile time.

Run time Storage Management

In the stack allocation, for each execution of a procedure a new activation record is pushed onto the stack. When the activation ends then the record is popped.

For the run-time allocation and deallocation of activation records the following three-address statements are associated:

1. Call
2. Return
3. Halt

Action, a placeholder for other statements

We assume that the run-time memory is divided into areas for:

1. Code
2. Static data
3. Stack

Run time Storage Management

Static allocation:

1. Implementation of call statement:

The following code is needed to implement static allocation:

MOV #here + 20, callee.static_area /*it saves return address*/</p>

GOTO callee.code_area /* It transfers control to the target code for t he called procedure*/

Where,

callee.static_area shows the address of the activation record.

callee.code_area shows the address of the first instruction for called procedure.

#here + 20 literal are used to return address of the instruction following GOTO.

Run time Storage Management

2. Implementation of return statement:

The following code is needed to implement return from procedure callee:

GOTO * callee.static_area

It is used to transfer the control to the address that is saved at the beginning of the activation record.

3. Implementation of action statement:

The ACTION instruction is used to implement action statement.

4. Implementation of halt statement:

The HALT statement is the final instruction that is used to return the control to the operating system.

Run time Storage Management

Stack allocation

Using the relative address, static allocation can become stack allocation for storage in activation records.

In stack allocation, register is used to store the position of activation record so words in activation records can be accessed as offsets from the value in this register.

The following code is needed to implement stack allocation:

1. Initialization of stack:

```
MOV #stackstart , SP    /*initializes stack*/
HALT                    /*terminate execution*/
```

Run time Storage Management

2. Implementation of Call statement:

ADD #caller.recordsize, SP/* increment stack pointer */

MOV #here + 16, *SP /*Save return address */

GOTO callee.code_area

- Where,

caller.recordsize is the size of the activation record

#here + 16 is the address of the instruction following the **GOTO**

3. Implementation of Return statement:

GOTO *0 (SP) /*return to the caller */

SUB #caller.recordsize, SP /*decrement SP and restore to p revious value */

Summarization