Q. 2] Explain how sensory instructions for vision system using VAL-II

→ Sensory Monitoring Commands :-

- Sensor signals & passing of sensor thresholds can invoke the execution of instruction sequence.

- Here the time of execution is ~~interro~~ unknown so that the main program is interrupted at any place and continued from afterwards

- However the instructions to the assigned sensors signals or threshold to program sections, for instance in content of general task concepts are rarely available.

- Sensor instructions for vision system :
Most vision system have three characteristics which aid their integration into robot programming system.

1. Parts to be identified can be assigned symbolic names repeated by ASCII strings.

2. Position & orientation are evaluated in relation to a cartesian co-ordinate system.

8 The system is taught that those parts to be recognized.

- The extension of VAL that is used for this sensory instructions for vision system is VAL-II.V contains several program commands.
  for a slave program which perform camera calibration, teaching & storage of features, set off various parts under symbolic names.

- The user program then communicate with the vision system by instructions 'V.PICTURE' & 'V.LOCATE'

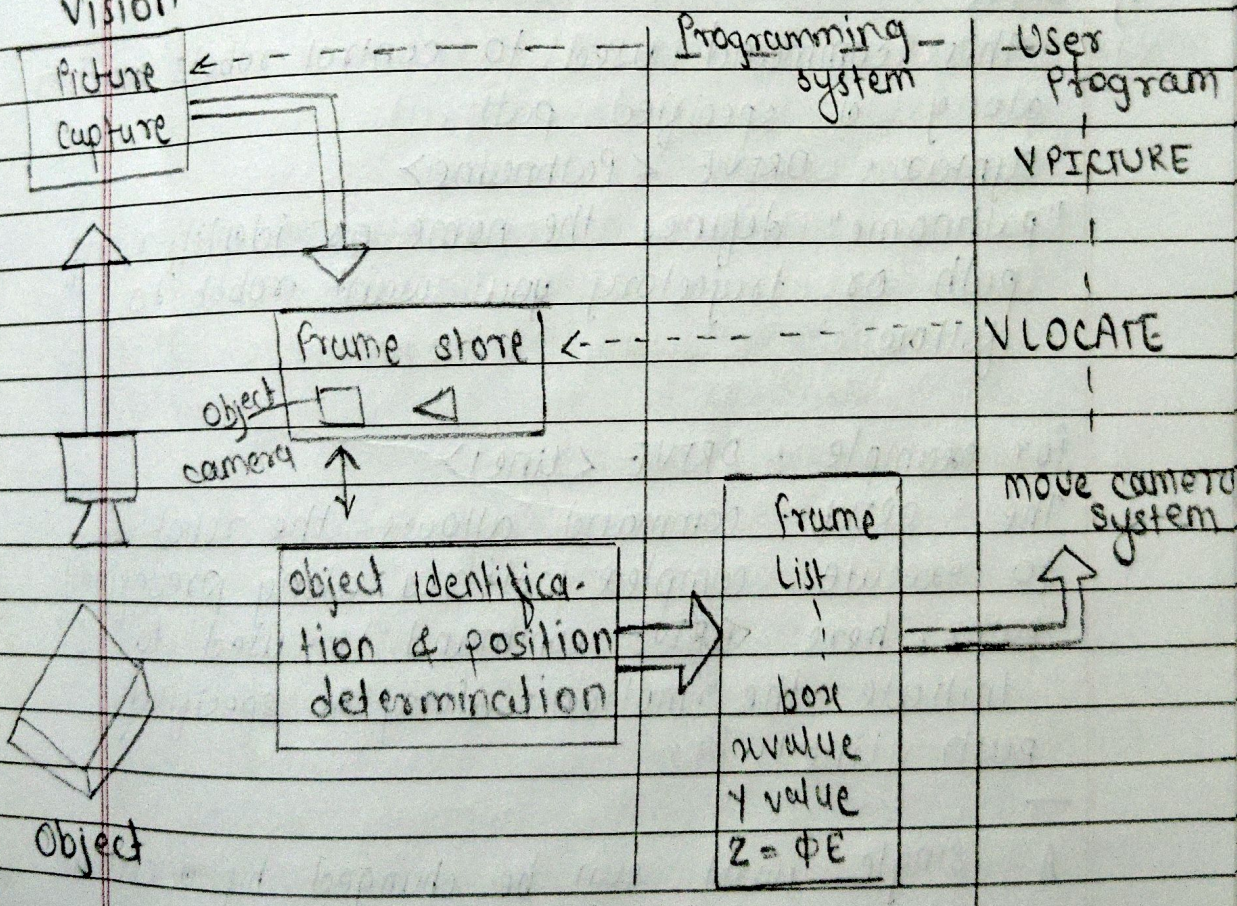
- 'V PICTURE' causes the capture of frame in the memory of vision system.

- V.LOCATE then activates the search for stored picture for a specified object In the search phase the program branches to a level if given or program stops output in an error messages.

- However if the search succeeds the vision system enters the position & orientation under a FRAME symbol which is same as object name.

specifying symbolic names for the parts identified by set of features which may contain area, number of holes, largest or smallest diameter, ~~of~~ circumference.

The user does not have to concern himself with setting up of this parameters the vision system automatically generate a set for the taught part & compare the stored values with set of features of current picture during program instructions.



Q.9] Explain the following instruction in VAL-II with example.

→

i] MOVET

In this the TCP moves along a transverse path.

syntax : ~~MOVET <P1>~~ MOVET <location>

for example

MOVET <P2>

The robot moves a path that is perpendicular direction to the robot's end effector that is P2.

2] DRIVE

This command used to control robots motion along a specified path.

syntax : DRIVE <Pathname>

"pathname" defines the name or identifier of path or trajectory you want robot to follow.

for example :- DRIVE <line1>

The DRIVE command allows the robot to execute complex motions along predefined paths, here DRIVE command is used to initiate the motion along a specified path line1.

—

A single joint can be changed by a certain amount with statement.

for ex :- DRIVE 4, -62.5

This command changes the angle of joint 4 by driving at it 62.5°.

iii] APPRO
This command gives instructions to robot to move near to point but offset from location along tool in z-axis.
syntax : APPRO <location>
for example :- APPRO P1, 50
The APPRO instruction are executed in joint interpolated motions. the instruction moves the tool away from P1 by 50 mm

iv] DELAY
Use to introduce pause or delay in execution of robot program. It allows us to specify a certain amount of time for robot to wait before proceeding to next instruction
syntax :-
DELAY 2
Gives DELAY for 2 seconds.

v] LISTP : It List out all programs.
for ex :- Program-A Size 10kb
Program-B Size 8kb
Program-C Size 12kb

vi] EXECUTE : It executes the given command
for ex :- EXECUTE Program-A
In this example EXECUTE command is used to execute program named 'Program-A'.

vii) RETRY : Used to repeat a block of code until a certain condition is met. Allows to create a loop in program

```
RETRY <CONDITION>
MOVE T
WAIT 1
ENDR
```

In this example <condition> represents the condition that needs to be evaluated the block of code between 'RETRY' & ENDR will be repeated until the condition specified ~~is~~ is met.

8] ENABLE :- The ENABLE command used to deactivate or enable specific functionalities or components of robot system
for ex:- ENABLE VisionSystem
Enable command activates the vision system of robot allowing it to perceive & proces visual info.

9] GRASP
The above command causes the gripper to close immediately
For ex :- GRASP 20, 15
above code checks whether the opening is less than the amount of 20 mm. If the opening is less than the amount of 20mm the program branches to statement 15.

10] LIST :
It list out all location or point
for example :
1. Home Co-ordinates (0,0,0)
2. Waypoint_A Co-ordinates (100, 50, 20)
3. Target Co-ordinates (200, 100, 30)
the output indicates there are three
location points.

11] PCABORT
The command is used to abort the current
program execution & stop the robot's motion
immediately.

12] RENAME
The RENAME command is used to rename a
program or label within a program
It allows for renaming existing programs
or labels to improve code readability
or to reflect changes in program
structure.

13] DISABLE
The VAL-II for robot programming is used
to deactivate specific functionalities.
for example :
DISABLE Visionsystem
The DISABLE command is used to manage
various components of robot system.

Q.4] Explain program instructions used in VAL-II

1. MOVE P1
This causes the robot to move in joint interpolation motion from its present location to location P1.

2. MOVES P1
Here the suffix s stands for straight line interpolation motion.

3. MOVE P1 VIA P2
This command instructs the robot to move from its present location to P1 passing through location P2.

4. APPRO P1 10
This command instructs the robot to move near to location P1 but offset from location along the tool z-axis in negative direction by a distance of 10.

DEPART 15
Similar to APPRO this instructs the robot to depart by a specified distance from its present position.

6] DEFINE PATH 1 = PATH (P1, P2, P3, P5) MOVE PATH 1.

The first command [DEFINE] defines a path that consist of series of location P1, P2 P3 and P5. The second command MOVE instructs the robot to move through these ~~commands~~ points in joint interpolation.

7] ABOVE & BELOW

These commands instructs the elbow of robot to point up & down respectively.

8] SPEED 50 IPS

This indicates that speed of end-effector during program execution should be 50 inch per second.

9] SPEED 75

This instructs the robot to operate at 75% of normal speed.

10] CLOSE

Instructs the end effector to close during the execution of next motion.

11] OPENI

causes the action to occur immediately.

12] CLOSEI

causes the action to occur immediately

If a gripper is controlled using a servo-mechanism the following command may also be available.

1] CLOSE 40mm

The width of finger opening should be 40 mm.

2] CLOSE 3.0 LB

This causes 3lb of gripping force to be applied against the part.

3] GRASP 10, 100

This statement causes the gripper to close immediately & checks whether the file opening is less than specified amount.

4] SIGNAL 4 ON

This allows the signal from output port 4 to be turned on at at one point in program.

5] WAIT 10 ON

This command makes the robot to wait to get signal on line 10 so that the device is on there.

Q.5] Explain various instructions motion in VAL-II

→

① MOVE ② APPRO ③ SPEED ④ MOVES
⑤ DEPART these instructions are explained in before question.

⑥ DRIVE 4, -62.5, 75
This command changes the angle of joint 4 by driving it 62.5 in -ve direction at a speed of 95 % of monitor speed.

⑦ ALIGN
This causes the tool to be rotated so that the z-axis is aligned parallel to the nearest axis of world co-ordinate system.

⑧ DO ALIGN
It is useful for lining up the tool before a series of locations are taughts in which it is important that tool be properly oriented.

Q. 6] Explain various monitor commands instruction in VAL-II

→

① HERE P1

One of the defining point location method is HERE command. The operator uses the teach pendant to move robot manipulator to the position to be defined & uses command as follow - HERE P1

this command given in monitor mode defines variable P1 to be current robot arm location.

② WHERE

This command queries the system to display the current location of robot in cartesian world co-ordinates & wrist joint variables.

It also displays the current gripper opening if the robot is equipped with position-servoed hand.

③ TEACH

The TEACH command is used to record a series of location values under the control of record button on teach pendant.

- Each time the record button is pressed a location variable is defined & given the value corresponding to location

of robot at the instant the record button is pressed.

- Each successive location variable ~~is defined & given the value~~ is automatically assigned a new name.

- The assigned name is derived from name specified in command
  for example
  TEACH P1.
  The first recorded location variable is P1 the next is P2 & so on. It is possible to teach a complete motion path

- Other monitor commands
  for storing programs onto disks (STORE)
  - copying programs (COPY)
  - loading files from disks to computer memory (LOAD)
  - listing the file names contained on a disk (FLIST)
  - renaming (RENAME) or delete (DELETE) files

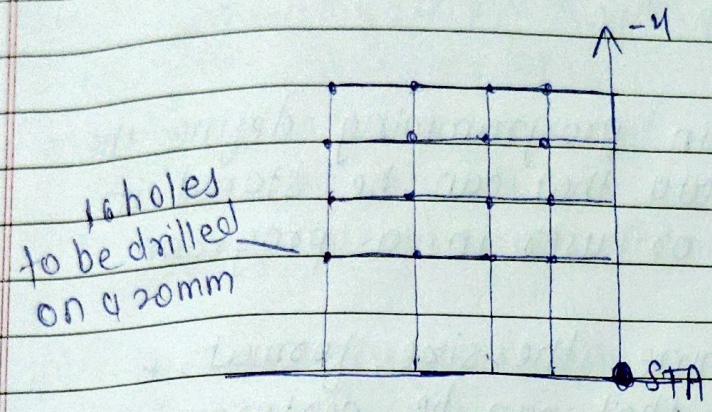
Programs

Q.7) Develop a program using VAL-II robot programming language for a PUMA 560 robot when setting output signal at 105th port & 5V voltage supply in controller it unlocals a cylindrical part of 10mm diameter from machine1 positioned at point P1 with co-ordinates (200, 250, 0)mm & orientation (0, 90, 0) and load the part on machine2 positioned at P2 with co-ordinates (200, 250, 50) & orientation (0, 90, 0) The speed of robot motion is 40 in/s However because of safety precautions the speed is reduced to 10 in/s while moving to a machine for an unloading or loading operation.

→

```
SIGNAL 105, 5
POINT P1 [200, 250, 0, 0, 90, 0]
POINT P2 [200, 250, 50, 0, 90, 0]
~~OPEN 100~~
SPEED 40 IPS
OPEN 100
APPRO P1, 50
SPEED 10 IPS
MOVE P1
APPRO P2, 50
SPEED 10 IPS
MOVE P2
BELOW
OPEN 100
DEPART P2, 50
STOP
```

Q. 8] Suppose we want to drill 16 holes according to the pattern shown in figure below.



Write a program for drilling in a pattern of grid as shown in figure above using VAL-II language. The speed during executing the motion will speed 20% of normal speed.

→
```
Main program
K=0
SPEED 20
FOR I = 1 TO 4
FOR J = 1 TO 4
k = k+1
CALL DRILL
END
END
TYPE "DRILLING COMPLETE"
Subroutine (DRILL)
XM = 20*I
YM = 20*J
MOVE SHIFT (STA BY XM, YM, 0)
DEPART 20
SPEED 10
DEPART 20
SPEED 20
```

Q.1 Define Data type. Explain any four data type used in RAPID language with the help of examples of programs.

→

Data types in programming define the kind of data that can be stored in a variable or user in a program.

They determine the size, format, & operations that can be performed on the data.

① num - Numeric values.

num is used for numeric values e.g. counters
- The value of num data type may be
  an integer e.g. -5
  a decimal number e.g. 3.45

It may also be written exponentially
e.g. 2E3

- Integers between -8388607 & +8388608 are always stored as exact integers.

ex :-

```
VAR num reg1 ;
...
reg1 := 3
```

reg1 is assigned value 3.

Limitations

There are literal values between -8388607 to +8388608 to a num variable are stored as exact integers.

② bool - Logical values

bool is used for logical values (true/false). The value of data of the type bool can be either TRUE or FALSE

example :-

flag 1 := TRUE ;

flag is assigned the value TRUE.

③ intnum - Interrupt identity.

intnum (interrupt numeric) is used to identify an interrupt.

- when a variable of type intnum is connected to a trap routine, it is given a specific value identifying the interrupt.
- This variable is then used in all dealings with the interrupt such as when ordering or disabling interrupt.

for example :-

- VAR intnum feeder_error;

...

~~Connect~~

CONNECT feeder_error with correct_feeder

ISIGNALDI dil , 1, feeder_error;

An interrupt is generated when the input dil is set to 1. when this happens a call is made to correct_feeder trap routine.

4. datapos - Enclosing block for a data object.

datapos is the enclosing block to data object retrieved with the function GetNextSym.

- Data of the type datapos contains information of where a certain object is defined in the system.

- It is used for instructions GetDataVal & SetDataVal.

- Basic examples of the data type datapos are illustrated below -

Ex :-

```
VAR datapos block;
VAR string name;
VAR bool truevar := TRUE;
...
SetDataSearch "bool" \Object := "my.**" \
InMod := "mymod" \LocalSym;
WHILE GetNextSym (name, block) DO
    SetDataVal name \Block := block, truevar
ENDWHILE
```

This session will set all local bool data objects that begin with my in the module mymod to TRUE.

5. dnum

dnum is used for numeric values
It can handle larger integer values than data type num bits but its characteristics and function is same for num.

Ex:-

```
VAR dnum reg1;
...
reg1 := 10000;
```

reg1 is assigned the value 10000;

Q.e] Explain position instructions and Input/Output Signal Instructions in RAPID with the help of examples of programs.

→

- Position Instructions

The robots movements are programmed as pose to pose movement, i.e. move from current position to new position the path between two position is automatically calculated by robot.

- Basic motion characteristics such as type of path specified by choosing appropriate position instructions.

- remaining motion characteristics are specified by data which are arguement or instructions;

position for robot & external axis)

Speed Data (Desired data)

zone Data (Position Accuracy)

Tool data (The position of TCP)

Work object data (current co-ordinate system)

alue of Analog & Digital signal is specified as a remaining value

Input & Output signal

vertDO - Inverts the value of digital output signal.

or example -

Invert DO 15;

The current value signal do15 is inverted

) BusStart - Start of I/O bus.

for example -

IOBusStart "IBS";

The instruction set start bus with name "IBS"

③ PulseDO

PulseDO

digital out

Example

A

generate

④ RESET

Example

⑤ Set

for

⑥ Set

⑦

③ PulseDO

PulseDO is used to generate a pulse on digital output signal.

Example :- PulseDO do15

A pulse with a pulse length of 0.2s is generated on the output signal do15.

④ RESET - Reset a digital output signal to zero

Example

Reset weld ;

The signal weld is set to 0 ;

⑤ Set - Sets a digital output signal to one.

for example :-

Set weldon ;

The signal weldon is set to 1.

⑥ SetDO - is used to change the value of digital output signal with or without a time delay or synchronization.

for ex :-

SetDO weld off ;

The signal weld is set to off.

⑦ SetAO - changes the value of analog output signal.

ex :- SetAO ao2 , 5.5 ;

The signal ao2 is set to 5.5.

8] SetGO - changes the value of a group of digital output signal.

Ex :- SetGO go2, 12;
The signal go2 is set to 12

Q.3] Explain motion command. Explain at least four move motion commands used in Rapid language with example.

→

- Motion commands in the context of robot programming refers to instructions or functions that control movement & positioning of a robot.

- These command enable the robot to perform specific motions in workspace

- These are move motion commands :-

① ~~MoveC~~ MoveC - Moves the robot circularly
MoveC is used to move the tool center point circularly to a given destination. During the movement the orientation can only be used in main task T_ROB1
for ex :-
MoveC p1, p2, v500, z30, tool2;
The TCP of tool, tool2 is moved circularly to position p2 with speed data v500 &

zone data z30. The circle is defined from start position, the circle point p1 & the destination point p2.

② MoveL - Moves the robot linearly.
It is used to move the tool center point linearly to a given destination.

When the TCP remain stationary then this instruction can also be used to reorientate the tool

for example :-
MoveL, p1, v1000, z30, tool2
The TCP of the tool is moved linearly to the position p1 with speed data v1000 & zone data z30.

③ MoveAbsJ - Moves the robot to an absolute joint position.
It is used for example
- the end point is a singular point.
- for ambigious positions on the IRB. 6400C

for ex :-
MoveAbsJ p50, v1000, z50, tool2;
The robot with the tool tool2 is moved along a non-linear path to absolute axes position p50 with velocity data v1000 & zone data z50.

④ MoveJ - Moves the robot by joint movement.

It is used to move the robot quickly from one point to another when the movement does not have to be in straight line.

for ex :- MoveJ p1, vmax, z30, tool2;

The tool center point of tool tool2 is moved along a non-linear path to position p1 with speed data vmax & zonedata z30.

Q.4] Explain functions. Explain any four functions used in RAPID with the help of examples of programs.

→

In programming, function is a named block of code that performs a specific task.

Functions provide a way to organize & modularize code making it more manageable, reusable & easier to understand.

① Asin - Calculates the arc sine value
Asin for ex:-

```
VAR num angle;
VAR num value;
;
angle := Asin(value);
```

angle will get the arc sine value

② ATan - Calculates the arc Tangent Value.
for ex:-

```
VAR num angle;
VAR num value;
;
angle := ATan(value);
```

angle will get the tan value.

③ AOutput - Reads the value of an analog output signal.
for ex:-

```
IF AOutput(a04) > 5 THEN...
```

If the current value of signal a04 is greater than 5 Then...

④ ArgName - is used to get the name of original data object for current argument
for ex:-

```
VAR num chales := 5;
proc1 chales

PROC proc1 (num par1)
VAR string name;
;
name := ArgName(par1);
```

TPWrite "Argument name " + name +
" with value " \Num := par1
END PROC

→ The variable name is assigned the string value "chales" & on flex pendant the following string is written "Arguement name chales with value 5".

5] Explain the following instructions in RAPID with the help of example of programs.

→ 1] AccSet
- Used when handling fragile loads.
It allows slower acceleration & decceleration which results in smoother robot movement.
for ex : AccSet 50, 100 ;
The acceleration is limited to 50% of normal value.

2] SetDO
It is used to change the value of digital output signal with or without time delay
for ex :-
SetDO weld off ;
The signal weld is set to off

3] MoveAbsJ
refer previous ques.

4] ISignalDO -
(Interrupt (Signal Digital output) is used to order and enable interrupts from a digital output signal.

for ex:-
VAR intnum siglint;
CONNECT siglint WITH routine1;
ISignalDO do1, 1, siglint;
Orders an interrupt which is occur each time the digital output signal do1 is set to 1.
A call is then made to iroutine1 trap routine.

5] WaitDO
Wait until a digital output signal is set.
for ex:-
WaitDO do4, 1;
Program execution continues only after do4 output has been set.

6] MoveL
It is used to move the tool center point (TCP) linearly to a given destination
-When the TCP is to remain stationary then this instruction can also be used to reorientate the tool.

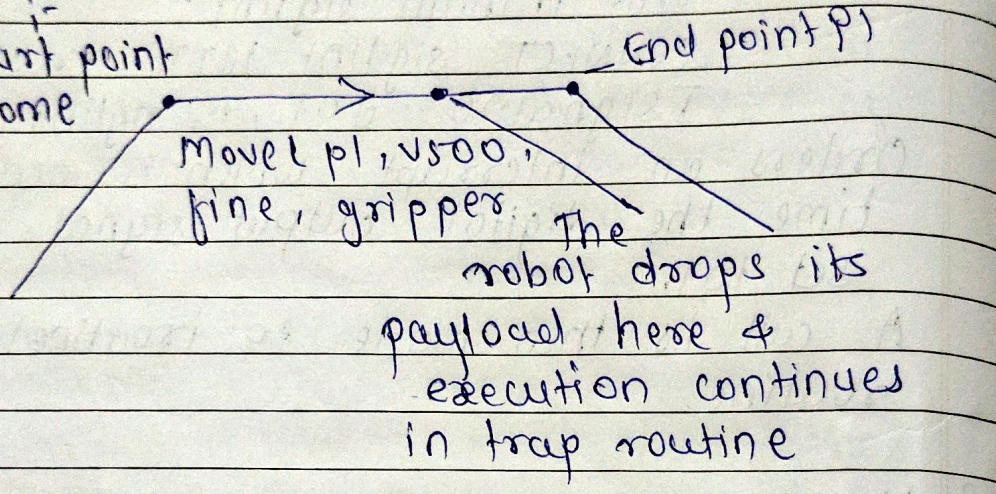
MoveL p1, v1000, z30, tool2;

The TCP of tool tool2 is moved linearly to the position p1 with speed data v1000, & zone data z30.

7] ClearPath

ClearPath clears the whole motion path on the current motion path level

ex :-




```
VAR intnum drop_payload;
VAR errnum ERR_DROP_LOAD :=-1

PROC minicycle ()
 BookErrNO ERR_DROP_LOAD;
 proc1 ;
 ERROR (ERR_DROP_LOAD)
 RETRY;
ENDPROC

PROC proc1()
   proc2 ;
ENDPROC
```

8] GripLoad

It is used to define the payload which the robots holds in its grippers.

for ex :-

GripLoad Piece1;

The robot gripper holds a load called piece1.

- when incorrect data is specified it can often lead to following consequences

- If the value in specified load data is greater than that of value of true load;
  ① The robot will not be used to its maximum capacity.
  ② Impaired ~~with~~ path accuracy including a risk of overshooting.

- If the value of in specified load data is less than the value of true load
  ① Impaired path accuracy including a risk of overshooting
  ② Risk of overloading the mechanical structure.

9] Triggl

Triggl is used to set output signals or run interrupt routines at fixed positions at the same time that the robot is making a linear movement

Ex : VAR trigg data gunon

TriggIO gunon, 0\Start\DOp:=gun1;

MoveJ p1, v500, z50, gun1;
TriggI p2, v500, gunon, fine, gun1;

The digital output signal gun is set when the robot's TCP passes the midpoint of corner path of point p1.

10] StopMove - Stops robot movement
StopMove is used to stop robot and external axes movements & any belonging process temporarily.

for example StopMove;
WaitDI ready-input, 1;
StartMove
The robot movement is stopped until the input is ready-input is set.

11] CONNECT
It is used to find the identity of an interrupt & connect it to a trap routine.
The interrupt is defined by ordering an interrupt event & specifying its identity.

```
VAR intnum feeder_low;
CONNECT feeder_low WITH feeder_empty;
ISignalDI di1, 1, feeder_low;
```

- An interrupt identity feeder_low is created which is connected to trap routine feeder-empty.

- There will be interrupt when input di1 is getting high. In other words, when this signal becomes high the feeder_empty trap routine is executed.

12] IDELETE.

IDELETE is used to cancel an interrupt subscription.

If the interruption is to be only temporarily disabled the instruction ISleep or IDisable should be used.

Ex :- IDELETE feeder_low

The interrupt feeder_low is cancelled.

Q. 6] Differentiate between TRAP Routine & PROC Routine in RAPID Language.

→

1. TRAP Routine

A trap routine (also known as a trap handler) is used to handle exceptions or errors that occur during execution of program.

- It is designed to capture and respond to a specific events or conditions that may disrupt the normal flow of program
- Trap routines can be defined to handle various types of exceptions such as hardware faults, software faults or specific user-defined conditions.

Differential parameters in TRAP routines refer to input/output parameters that are passed when the routine is triggered

These parameters allow the TRAP routine to access information about the exception or error occured and take appropriate actions to handle it.

For example :- differential parameters can include error codes, status flags or any other relevant data needed to diagnose & respond to exception.

2. PROC Routine

A PROC routine is a type of routine used to encapsulate a specific set of instructions that need to be performed.

PROC routines are commonly used for modular programming where a complex task is broken down into smaller, more manageable procedures.

These routines can be called from other parts of programs to execute the define set of instructions.

Differential parameters in PROC routines refer to the input/output parameters that allow data to be passed into & out of routine.

These parameter enable the PROC routine to receive input values from the calling program or provide output values back to calling program.

They serve as the means of communication between the calling program & PROC routine allowing the exchange of data & results.