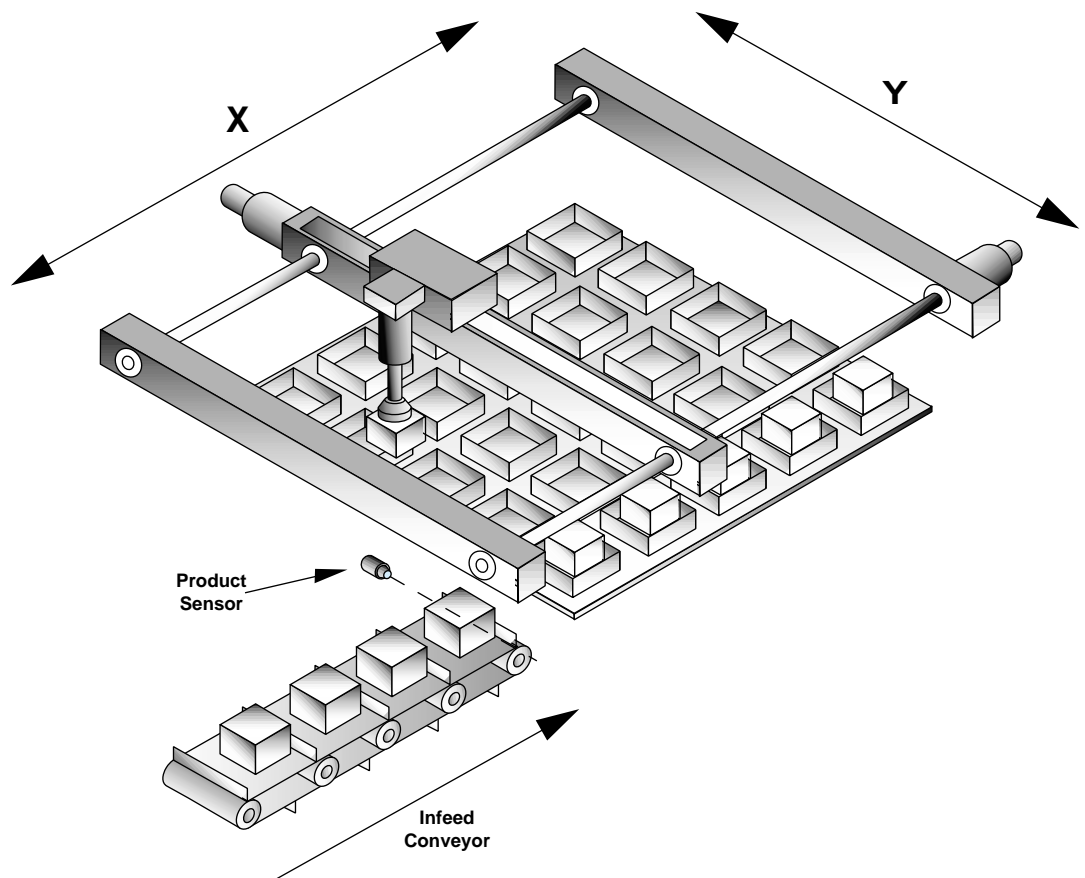


2 Axis Pick & Place System

A square palette has sides 1200mm long.

It must be divided into a grid, each of these positions on the palette contains a box into which a widget must be placed:

A vacuum operated pickup mechanism collects objects from a conveyor and fills each of the boxes in turn.



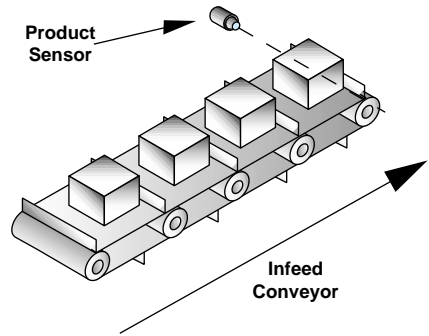
Additional Information:

Whist the palette size is fixed at the maximum size of 1.2m square, the program should be flexible enough to allow for a user-defined number of boxes on the palette. The grid could contain up to 10 divisions in each direction and they may be combined in any ratio, i.e 2x4, 6x 3, 1x10, 9x4 etc.

Infeed - Axis 2

The conveyor has fixed dividers 200mm apart with a sensor connected to input(4) to test for the presence of a product at the pickup point.

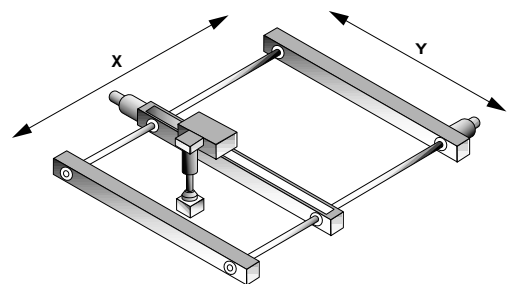
The conveyor needs to be indexed one pitch at a time and the pick & place mechanism synchronised to wait for the conveyor to be stationary before fetching the product.



X/Y mechanism - Axes 0 & 1

These axis are very simple. The units are in mm and a datum sensor is fitted at the zero position.

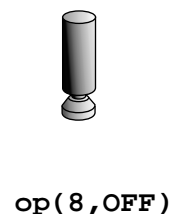
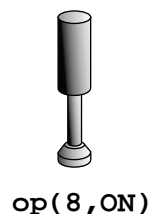
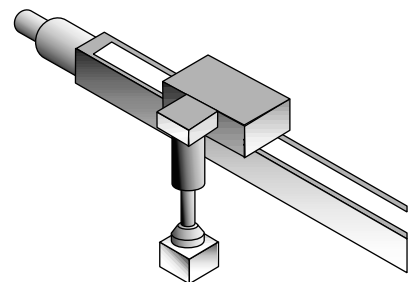
Providing that the pickup arm is in the raised position the X/Y axes can move freely over the entire palette area.



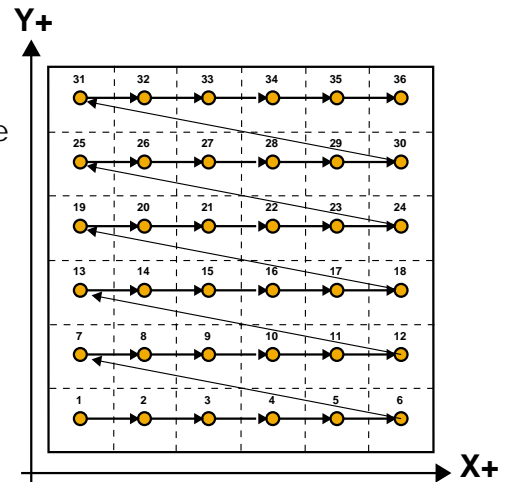
Z Axis Pickup Unit

This axis consists of a vacuum pickup unit attached to a pneumatic actuator.

The actuator is controller by a digital output (channel 8) ON = DOWN and the vacuum unit is switch via output(9).



The illustration on the right shows a sample palette with a 6x6 grid of boxes. The numbers/arrows show the order in which the boxes are filled. Note that we step through the rows (Y) in turn, filling each box (X+) before moving onto the next row.



Structuring the program

This example can be solved with a very simple structure using two nested FOR..NEXT loops.

Firstly we create a loop to step through each row (Y) in turn:

```
FOR y=0 to ydiv-1
NEXT y
```

'ydiv' is the number of rows, and the -1 is because we start counting at 0 rather than 1.

Now, within this loop we create another for the 'X' direction:

```
FOR y=0 to ydiv-1
  FOR x=0 to xdiv-1
  \
  NEXT x
NEXT y
```

Calculating the box positions

So now we have a sequence which steps sequentially through each row, and then through each position on that row in turn. We can use the absolute move (MOVEABS) command to position the axes at an absolute position in our X/Y coordinate system in the form

```
MOVEABS(x,y)
```

The x and y variables with the FOR..NEXT loop are simply logical box coordinates and therefore need to be scaled to the correct positions.

If we know the palette size (1200) and the number of divisions in each direction (xdiv/ydiv) then we can simply calculate an appropriate scaling, thus for the x axis:

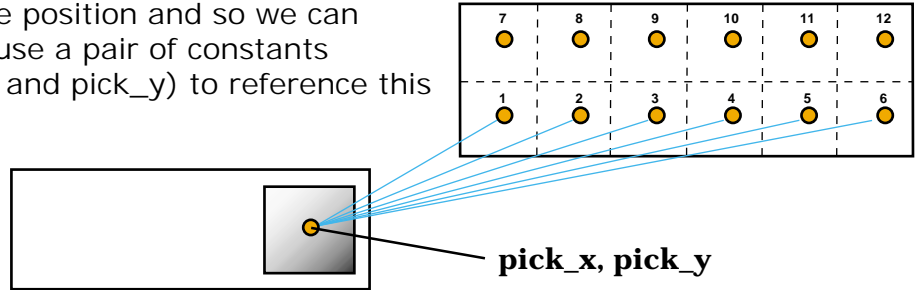
```
xscale = 1200/xdiv
```

The actual position (of the boxes corner) would therefore be (x*xscale), we could adjust this for the centre of the box by adding half the box size (xscale/2). So the position would be:

```
(x*xscale)+(xscale/2)
```

Our final consideration is that for each box we first have to move to the preset pick-up position, fetch the product and then move to the appropriate empty box to place the product in.

The pick up point is at a known absolute position and so we can simply use a pair of constants (pick_x and pick_y) to reference this point.



Code Example

```

constants:
  nozzle=8
  vacuum=9

  xdiv=4
  ydiv=5

start:
  xscale=1200/xdiv
  xmid=xscale/2
  yscale=1200/ydiv
  ymid=yscale/2

  FOR y=0 TO xdiv-1
    FOR y=0 TO ydiv-1

      GOSUB pick

      MOVEABS( (x*xscale)+xmid,(y*yscale)+ymid )
      WAIT IDLE
      GOSUB place

    NEXT x
  NEXT y
  GOTO start

pick:
  MOVEABS(pick_x, pick_y)
  WAIT IDLE

  OP(nozzle,ON)
  OP(vacuum,ON)
  wa(500)
  OP(nozzle,OFF)
  wa(500)
  RETURN

place:
  OP(nozzle,ON)
  OP(vacuum,OFF)
  wa(250)
  OP(nozzle,OFF)
  wa(500)
  RETURN
  
```