

## Multi-Tasking: Communicating Between Tasks

The multi-tasking capability of the Motion Coordinator Series is a very powerful feature, we do however need a mechanism to enable us to synchronise events between multiple tasks and to pass information between them.

The simplest way to achieve this is to use either the global variables (VR()'s ) or flags. In principle, all we need to do is to set a flag, or write to a variable in one task, and then wait for that change to occur in another.

Only one task should write to any particular variable (though of course any task can read it). A separate variable could be used to acknowledge an event has occurred.

When one task is waiting on an event in another task, a time out is advisable rather than simply 'busy waiting'

```

: main_lp:
○ . \
: . \   Body of code
○ . \
: . TICKS = 5000
○ . REPEAT
: . UNTIL ((VR(wrap_complete)=1 ) OR (TICKS <=0))
○ .
: . IF TICKS <=0 THEN
○ .   GOTO error_timeout
: . ELSE
○ .   GOSUB transfer
: . ENDIF
○ .
○ . GOTO main_lp
○ .

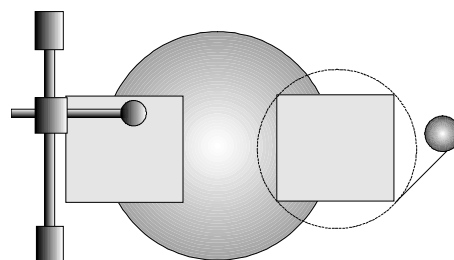
```

### Example:

A pick & place machine is filling boxes on a palette. When the palette is full a transfer carousel will replace the palette with a fresh one and a secondary operation will spiral-wrap the whole palette with plastic film.

We can assume that the palletising operation is a modified version of the program generated in the X/Y pick & place example.

Top View



Side View

