Denford

DENFORD

COMPUTERISED MACHINES AND SYSTEMS

Computer Integrated Manufacturing - CIM Manual

Denford Limited
Birds Royd
Brighouse
West Yorkshire
England
HD6 1NB
Tel. +44 (0)1484 712264
Fax. +44 (0)1484 722160
email: info@denford.co.uk

Contents

Introduction

In the past few years, world-industry has changed its approach to Manufacturing with the development and use of Computer Integrated Manufacturing (CIM) systems. There has been an emphasis on the *integration* of engineering, design, manufacturing-analysis, and business management, in an attempt to create an enterprise wide, co-ordinated manufacturing system. As new manufacturing methods and new technologies emerge, it is essential for the continued development of CIM to produce Engineering Graduates with both a sound base in the theory of CIM and real, hands-on, experience with the types of systems currently in use. To help continue the research into CIM and to provide an industrial standard training facility, Denford's Systems Division has developed new software, which, combined with a wide range of hardware devices and the best third party software, will enable a comprehensive CIM system to be brought into a teaching environment.

The Denford CIM

Throughout the project development, four major design criteria have been adhered to. The first and most important consideration was that the software had to be *completely flexible*. A wide range of machine tools and automation devices are available; all of which must be able to integrate with the software. Using the Open Systems principle of sharing data, the Denford system can interface to other, third-party software such as Computer Aided Design (CAD) and Computer Aided Manufacturing (CAM) packages, databases and other manufacturing or business management software.

The second is that the system should be of reasonably *low cost*. This is achieved through the use of PC compatible computers and the Microsoft Windows operating environment. These two components are already established as standards in both industry and education and provide a low-cost, yet extremely powerful foundation for the CIM software.

The third design criterion was to make the software as *modular* as possible. The software is configured as a set of programs each performing a distinct task and storing results in a central database. This allows each software module to either integrate into the rest of the system or perform as a stand-alone program. Each module can also be switched to perform a simulation completely off-line from the manufacturing system.

The fourth requirement was that the system had to be *expandable*. This is achieved through the modularity and the flexibility of the system. It allows an initial purchase of a skeleton system with expansion when more hardware and

software is purchased. The expansion of the system is relatively simple and could be carried out with no technical help from Denfords.

Software Architecture

The CIM system has been designed as a three level hierarchy, from the machine tools on the shop floor at the bottom level, up to the system supervisor the Host computer at the top. This structure was chosen since it allows effective delegation of responsibilities from the top down and timely feedback of relevant information from the bottom up.

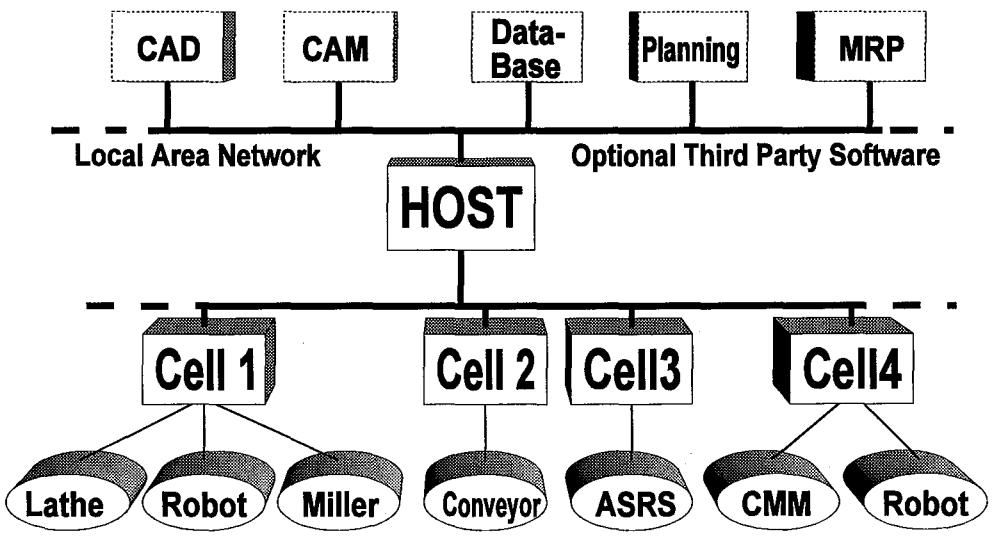


figure 1

The Host

The Host is usually a separate computer connected to a Local Area Network (LAN). Its primary role is to be the main user-interface to the manufacturing process. To this end it is composed of a number of software modules for the off-line set up of the shop floor, the design and route-planning for components and the high level control, monitoring and simulation of the entire manufacturing process.

The Host acts as the central file server to the set of cells. It dispatches information upon initiation of a process and receives status information, which it stores in a journal. The other important role of the Host is to interface to third-party software, either running on the same computer as the host software or across the network.

The Cells

A Cell is defined as a set of device-drivers, each responsible for the direct control of one machine and a low-level scheduling module to co-ordinate the actions of the machines. A cell has a local file-store, and can retrieve information from the host when required; therefore the system's database may be said to be distributed. An important element of the cell controller software is to maintain a list of interlocks, or safety checks, and regulate all machine actions for potential danger.

Connecting the Host and Cells in the top two levels in the hierarchy is a Local Area Network (LAN), which may be part of a larger Network (for example, Novell), or may simply be contained within the shop floor. The lower level machine tool connections are governed by the characteristics of the tools themselves but are likely to be RS232, RS485 and I/O lines. The Host may also interface to software above the shop floor control hierarchy, via the LAN.

The hierarchical structure ensures that there is a clearly defined distribution of functions from the host computer down to the machine tools themselves. For instance, if the complete schedule for production of a batch of different components were to be calculated and dispatched from one computer process, that process would be very slow and inefficient. By breaking down a problem such as scheduling into different levels of reduced complexity, the parallel processing power of multiple processes running on multiple PCs helps to increase the efficiency and flexibility of the software.

The Host

The host is the CIM supervisor, positioned at the top of the shop floor control hierarchy. It can communicate directly with all of the cells and the LAN but is unaware of particular machine and communication specifications. The Host acts as the interface between the user and the entire manufacturing process. In the context of the hierarchical architecture, it is the function of the host to initialise, control and monitor the set of cells in the layer beneath it.

To initialise the cells, a set of off-line configuration utilities are provided for shop floor design and analysis. The control and monitoring functions are performed using a real-time process scheduler called a *Dispatcher*. Packets of information are sent by the Dispatcher to the cell-controllers, which initiate sequences of machine actions. The real-time events of the machines are controlled and monitored by the cell controller, and the real-time events of the cell controller itself are communicated directly to the Dispatcher.

This chapter will first cover the off-line software modules and then describe the real time functions. Each software module uses the Windows environment of pull-down menus, graphical buttons and icons. These instructions make no attempt to explain the standard Windows features, but concentrate on the functionality and the interfaces of the Denford CIM Host software modules. To learn more about the graphical interface elements of Windows itself, consult your Windows documentation.

The Simulator

The first part of the simulator module is for designing and analysing a shop floor layout. Using the mouse, machine tools can be chosen, moved into place and then given a name and characteristics. The shop floor can be re-designed to suit the particular machine tool installation, or made up to show a completely simulated factory.

A range of machine tools may be included in the shop floor layout to do any type of job. The default set is for metal-working manufacturing and includes: Lathes and Millers of different sizes; different types of Robot; Measuring and Inspection devices; Automatic Storage and Retrieval Systems (AS/RS); Material Hoppers; Conveyors and Automatic Guided Vehicles (AGV). These machines are grouped into functional categories such as 'Tool', 'Inspection' and 'Transport' devices. This helps in the design and cellular analysis of the shop floor.

As with a real system, the user is free to design the particular Conveyor layout using modular conveyor sections that match the available hardware. Similarly the AGV route can be defined and altered to suit the particular shop floor design.

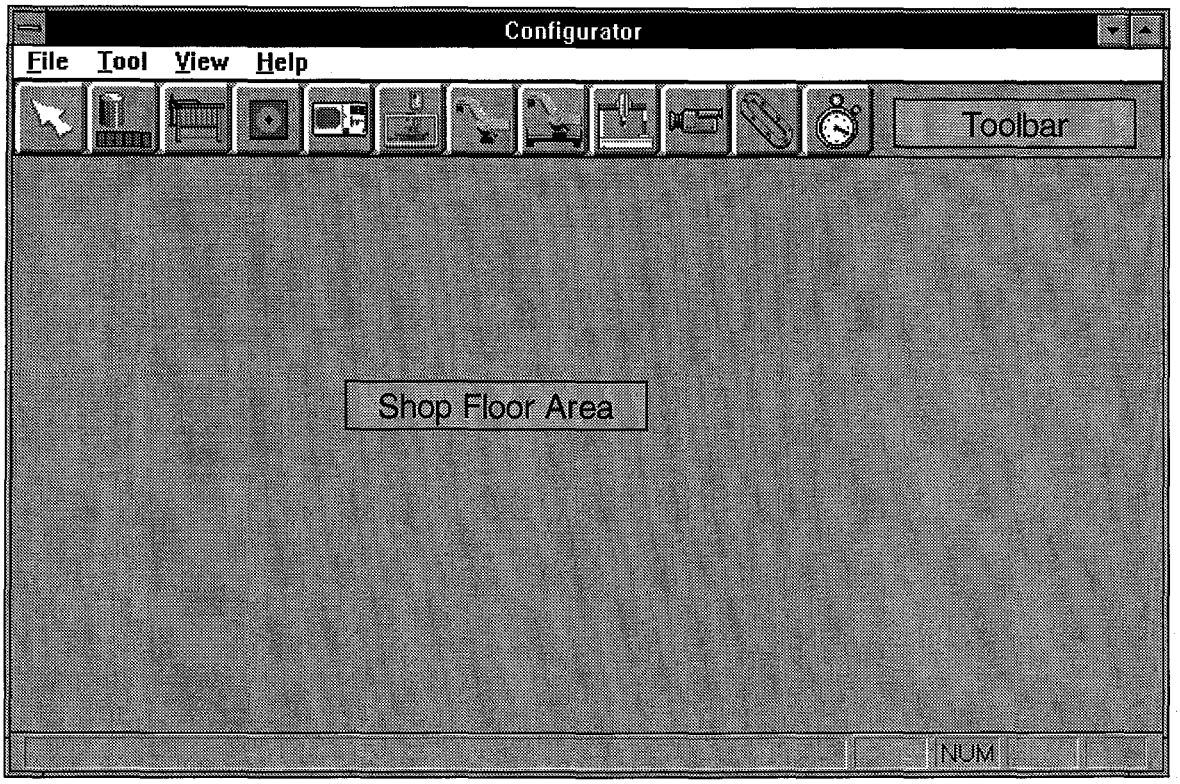
When the shop floor layout is completed, the user may calculate the cellular structure of the shop floor. The software groups machine tools into autonomous cells and displays the result to the user. These cells may be altered according to the user's needs.

A further analysis tool can be used to show the connectivity of the machine tools on the shop floor. A Transportation Network can be created which shows how material is moved about the shop floor.

Usage

When the Configurator is started, the main screen is displayed, showing the menu and the tool-bar which contains the current selection of available machine tools.

The window initially shows a new, empty shop floor. While the Configurator is running, a new shop floor can be created at any time using the **New** item in the **File** menu.



The Configurator's Main Window

The cursor is a white arrow when moved inside the Shop Floor Area. When the mouse is moved over the Toolbar, it changes to the standard Windows cursor. When a machine tool button is pressed, the cursor will alter shape so that it shows which tool is currently active; note, the cursor only shows the active tool when it is within the shop floor area. If the 'lathe' button were pressed, the cursor would become a small, black and white, lathe shape.

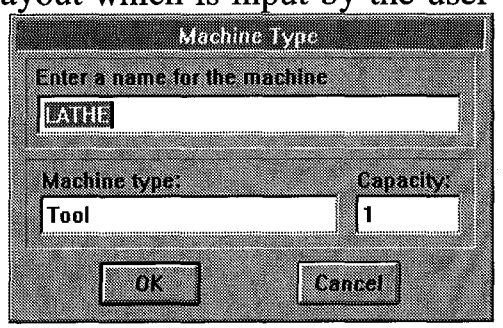
To position a lathe on the shop floor, press the lathe button and move the cursor to the required position in the shop floor area. Press and hold down the left mouse button; a rectangle showing the exact dimensions of the lathe appears which may be moved around by continuing to hold the left button down whilst moving the mouse. When the left mouse button is released, the lathe is finally drawn and a dialogue box appears requesting the input of a unique name for this machine tool. When a name is typed and the dialogue box is closed, the cursor remains as a lathe in case more than one of that particular machine tool is required. The user is free to change the cursor to another machine tool or to the arrow by pressing the appropriate button on the toolbar.

If the arrow is selected, the machine tools on the shop floor can be moved around. A machine tool on the shop floor can be *selected* by clicking the left mouse button on its icon on the shop floor window. A tool may be dragged to a different position by holding down the left mouse button over the machine tool icon in the shop floor window. When the mouse is moved with the left button

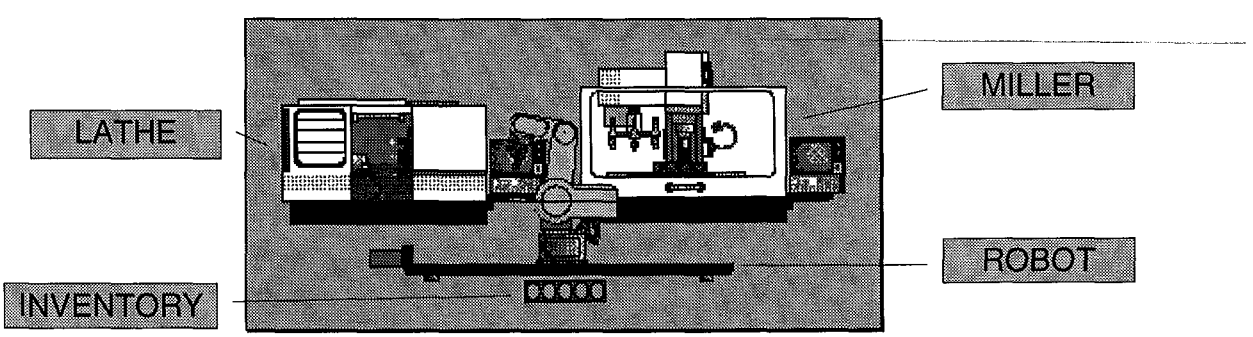
depressed, a rectangle showing the tool's dimensions moves across the shop floor. When the mouse is released, the machine tool assumes its new position and remains selected.

When the arrow tool is selected, a user may double-click over a machine tool icon in the shop floor window. This action causes the machine tool dialogue box to be displayed, the same that was displayed when the machine was initially placed on the shop floor layout. If the user double-clicks over a conveyor, the conveyor design utility will be started to allow the user to edit the layout, see the conveyor design utility section later on.

The dialogue box shows three pieces of information, the first being the name of the machine tool in this particular shop floor layout which is input by the user and may be anything which is meaningful to the user. The Machine type shows what class the selected machine fits into: machine classes are *Tool*, *Robot*, *Transport*, *Inspection*, *Misc*, *Station* and *Inventory*. The lathe chosen in this example is of type Tool. The last box shows how many components the device can handle at any one time: the capacity. In the case of this lathe, it is assumed that it has a work-holding device capable of holding a single component. This figure may be changed according to the type of machine and its proposed purpose.



Using the methods outlined above, a simple manufacturing cell may be created. The cell illustrated here consists of a lathe, a milling machine, a robot and a simple inventory device. Note, each shop floor must have an inventory device of some sort, since any form of manufacturing requires a supply of raw material.



Once the shop floor is defined, the user can analyse some important features, namely the shop floor's 'cellular structure' and 'transportation network'. The cellular structure of a shop floor is how the layout is broken down into manufacturing cells. The configurator defines a cell as an autonomous unit of

Configurator Controls

The File Menu

New – Creates a new, empty shop floor window.

Open – Retrieves an existing shop floor from the database and displays it in a new window.

Save – First ensures that all data-structures have been generated and prompts the user if they have not. If a shop floor is untitled, a dialogue box requests a name. If the data structures are complete, then the three data structures will be saved to the database with the following extensions:

.sfl The shop floor layout

.cel The cellular structure

.net The transportation net

SaveAs – Allows the shop floor to be saved with a different name.

Exit – Quits the application after first making sure that all data is saved.

The View Menu

ShopFloor – Shows the basic shop floor and allows the user to edit the machine layout.

CellStructure – Calculates and displays the Cellular structure of the shop floor.

TransportNet – Calculates and displays the hierarchical transportation network for materials in the shop floor.

Robot Reach – If a robot is currently selected, this menu item opens a dialogue box displaying the *reach envelope* for this robot (the area around it that can be reached by the gripper). It also shows which other machine tools are situated within the area and can therefore be loaded or unloaded by that robot.

The Toolbar

Along the top of the main window of the Configurator is a row of buttons called the toolbar. The buttons show icons of different machine tools and allow the user to select the different machines to place onto the shop floor.

The toolbar is created when the Configurator is started from an Initialisation file called **config.ini**, found in the **\cim\bin** directory. This file is structured to allow machines to be grouped under different categories according to their functions. The categories are:

Inventory – A machine or device to store raw materials, completed components or both. It may be automated or a simple rack-like device to be used with a robot.

Station – An transfer point between cells, either a physical device such as a palette-stop on a Conveyor, where material is transferred from the Conveyor Cell to another cell, or a logical transfer point where material simply moves from one cell to another. For example, an Automated Storage and Retrieval System loads palettes onto a Conveyor at a logical transfer point because there is no physical device to act as an interface.

Tool – A machine tool which alters the shape or appearance of some material or partially completed component, such as a Lathe, Miller, or De-burring machine.

Robot – Most often an 'in-cell' material handling machine used for pick-and-place operations, but could be used for 'inter-cell' transport, assembly, or component feeding from an inventory device.

Inspection – A device which reports the status or measurements of a component to the supervising Cell-Manager.

Transport – An 'inter-cell' material transport device such as a Conveyor or an Automated Guided Vehicle.

Misc. – Anything not covered by the above categories. Note that the 'Delay' machine falls into this category; it may be used as a communication relay to another device. For more details see the Delay Machine in the following section.

In the Initialisation file, the categories are enclosed in square brackets and are followed by a list of machine tool entries, each with numbers which internally reference the machine and its corresponding button in the toolbar. These numbers are pre-set for the default set of machine tools.

The Default Machine Tool Set

Inventory

 The Automated Storage and Retrieval System (AS/RS) comprises a gantry robot arm and two banks of storage racks, one for storing raw materials and one for completed components.

 A completely un-automated storage rack for material, used in conjunction with a pick-and-place robot capable of reaching each component in the rack.

Station

 The interface between cells where a palette or a component leaves one cell and enters another.

Robot

 A simple pick-and-place articulated robot, suitable for load/unload operations with machine tools. It can also be used with a simple Inventory device.

 The same robot as above mounted on a sliding base to increase its reach; the robot can now service more than one machine.

Transport

 This icon starts the built-in Conveyor Design Utility which allows you to define a conveyor layout (indexible or palletised) and place it on the shop floor. The Conveyor Design Utility is described in the next section.

 The Automated Guided Vehicle (AGV) tape icon allows the user to create a path for the AGV. See below for a description of its use.

Tool

 A small CNC lathe suitable to be used in an automated environment, i.e. with an automated guard and a tool-changer.

 A small CNC milling machine suitable to be used in an automated environment, i.e. with an automatic guard and tool-changer.

Inspection

 A Co-ordinate Measuring Machine (CMM) for accurate and detailed measurements of components. It must be serviced by a robot.

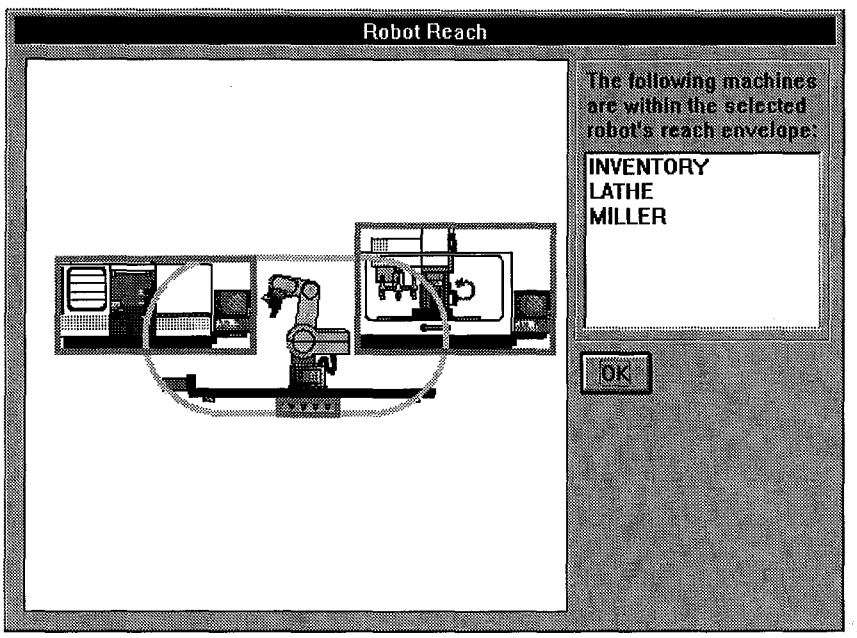
 A video camera which can be mounted alongside or above a station. It may be used for a variety of purposes including material verification, component recognition or measurement.

Misc.

 The Delay Machine. The final device category includes the clock icon, which may be used at a station to stop a palette or component for a period of time. This device may also be used to send a command to any other cell in the system. See the case study for an example of its usage.

Robot Reach Utility

When the user selects a robot on the shop floor, the **Robot Reach** item from the **View** menu can be used to show graphically the robot's reach envelope.



The area to the left shows the robot's envelope and all machines nearby. Each machine within that envelope is hi-lighted in green and the machines are listed on the right. Pressing the **OK** button returns to the Shop Floor view.

mouse button down. When the button is released the conveyor section is finally positioned in the design area.

A conveyor section or station can be removed using the Segment Delete button. When the cursor moves in the design area, a highlight appears around the conveyor section below the cursor. If the mouse button is pressed, that section is removed. Again, the Segment Delete function remains active until a conveyor section button is pressed.

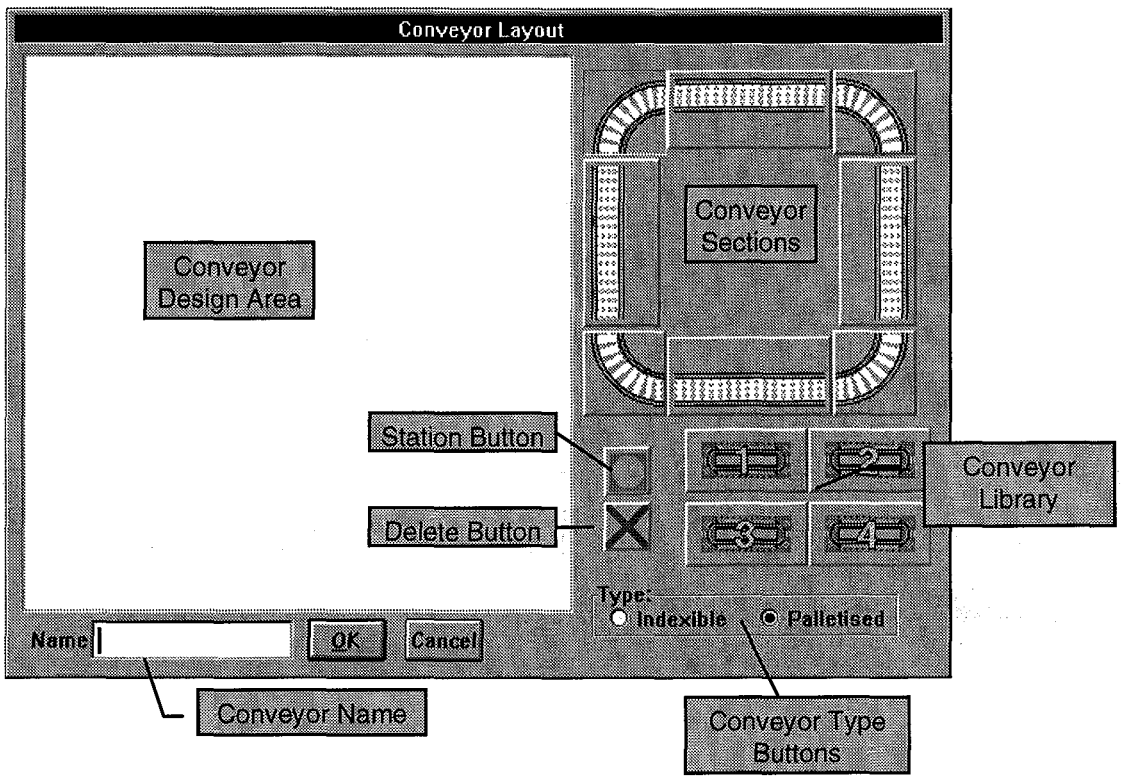
There are two primary types of conveyor available: an indexible conveyor, on which components move according to explicit commands from the conveyor's device driver, and a palletised conveyor, where components in special work-holding palettes move continually on the conveyor and must be stopped at a station using a physical stopping device. The conveyor selection buttons at the bottom of the dialogue box indicate the conveyor type. When the type is changed, the design area is cleared and the conveyor section graphics change to show the new conveyor type.

It is important to note that the direction of flow on the conveyor has to be consistent within a single conveyor layout.

When the conveyor is complete it must be given a name in the **Name** edit box. This name must be no more than eight characters long since it is used as a separate filename when the shop floor is saved. The conveyor is saved to a file with a **.dev** extension. When the **OK** button is pressed, the dialogue is closed and the cursor appears as a conveyor shape over the shop floor area. The new conveyor can be placed anywhere on the shop floor in the usual manner.

The Conveyor Design Utility

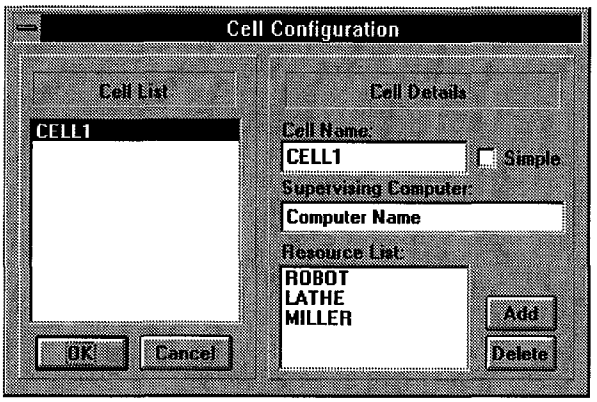
When a conveyor is integrated into a shop floor layout its shape must be defined to ensure that material and components flow as desired to each cell. Modern conveyors tend to be constructed in modular sections, e.g. straight sections, curved sections and corner sections. The software in the Configurator reflects this and allows the user to design a customised conveyor layout. Pressing the Conveyor button in the toolbar produces a dialogue box which allows the user to define the shape of a conveyor.



The buttons on the right represent the modular conveyor sections available. Unlike the toolbar buttons in the main window, a conveyor-section button remains depressed and stays 'selected' until the user chooses a different button. This allows the user to place several identical sections without continuously pressing the same button. Stations can also be added where pallet stop devices are situated on a palletised conveyor or where material enters or leaves an indexible conveyor.

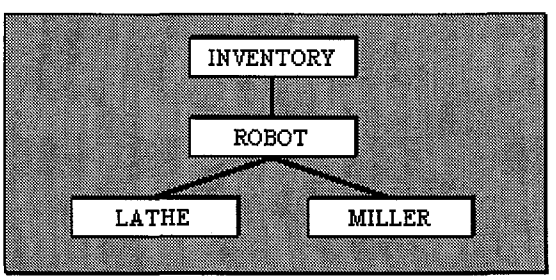
To place a conveyor section or a station into the design area, a similar procedure to placing a machine onto a shop floor layout must be performed; select a conveyor section by pressing the appropriate button, move the mouse to the design area (note the cursor changes to a screw-driver) and press and hold the left mouse button down. A rectangle showing the dimensions of the conveyor section appears and can be moved around whilst holding the left

automation capable of performing a single task, such as machining, material transport, storage, inspection and so on. The example of the lathe, miller, robot and simple inventory device constitutes a single machining cell, as can be seen using the **View** menu and selecting the **CellStructure** item.



Using the Cell Structure command, the user can analyse the shop floor to inspect the cellular structure. The left side of this dialogue box shows a list of all the cells in the system. If this is the first time the cell structure has been calculated, the computer generates names for the cells. A cell may be selected from the **Cell List** by clicking the left mouse button over its name. When a cell is selected, its details are shown in the right half of the dialogue box. The Cell Name and Supervising Computer can be changed by altering the text in the edit box. The **Resource List** – the machines contained within the cell – can be altered with the two buttons labelled **Add** and **Delete**. When adding a machine, a dialogue box appears which lists all the machines on the shop floor. One machine can be selected from this list and the **OK** button returns to the Cell Configuration dialogue and inserts the machine into the Resource List. To delete a machine from the Resource List, select it and press the **Delete** button.

The transportation network is another important analysis tool. This allows the user to see how material flows throughout the system, starting from the inventory device.



This network can be calculated and displayed using the **View** menu, by selecting the **TransportNetwork** item. The network above corresponds to the single cell demonstration previously given and shows the material transport from its inventory device, through material handling devices and into machines.

Three important data structures were generated and displayed in the above process: the shop floor layout; its cellular structure; and the material transportation network. To save these to the database, choose **Save** from the

File menu and type in the name for this shop floor. Three files will be generated with the extensions **.sfl**, **.cel** and **.net** corresponding to the data structures above. If the user saves a Shop Floor Layout without first viewing the Cell Configuration or the Transport Network, the data structures are generated and saved automatically.

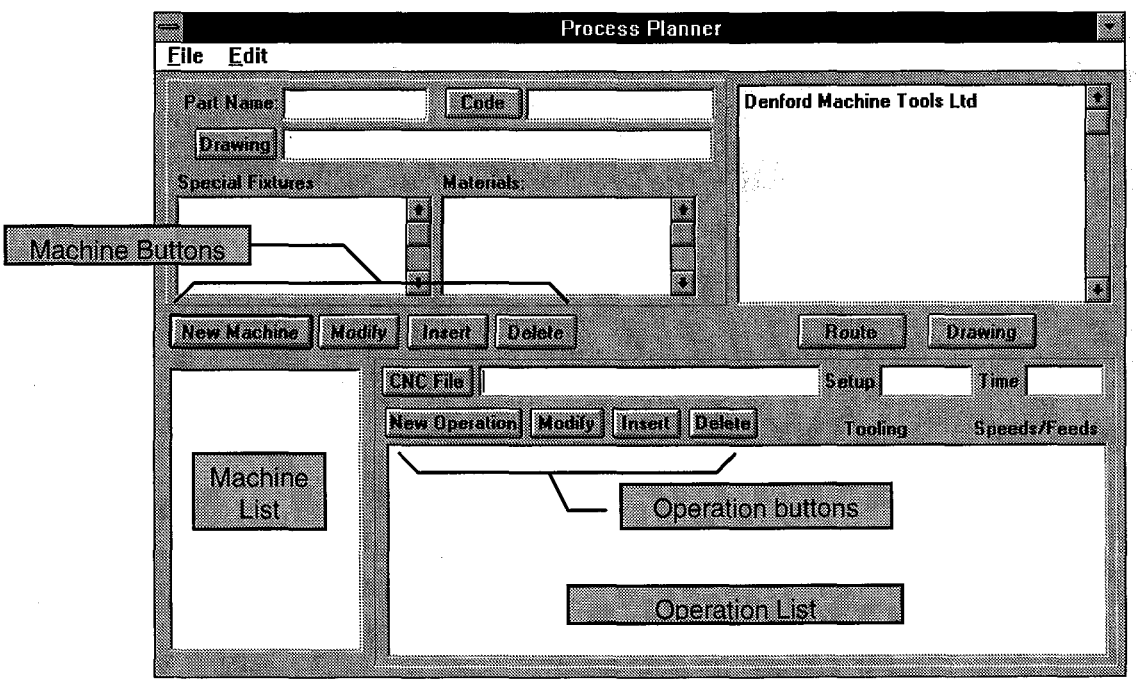
The Process Planner

The Process Planner is simply a documentation tool, used to describe the functions necessary to produce a component. Each component has a unique name, an identification number and several other properties defined by the process plan; the machining and measurement requirements are listed together with timing information and all tooling and work-holding data.

To enhance the power of the Process Planner, the Opitz Component and Material Classification System is provided to help apply a group-technology identification number to each component. Each digit of the Opitz-number refers to a particular characteristic of the component, hence components with similar characteristics can be grouped into component families.

Usage

The Process Planner is a documentation tool which aids in the creation of a component database. Its interface is a simple form to be filled in with the relevant details. When the application is started, a blank form appears:



The Process Planner

To create a new component the component name must be entered into the **Part Name** box. All other information is optional, but completing the form provides a useful record of the manufacturing requirements of the component.

The **Code** box may hold any internal reference to a component which is meaningful to the user's organisation, or may be left blank. Another option is to generate a group-theory component code by pressing the **Code** button. ()

A CAD drawing may be associated with the component by typing its name into the **Drawing** box, or by using the button to select one from the database. The **Drawing** button creates a standard file selection dialogue and transfers the chosen name to the Drawing box on the form.

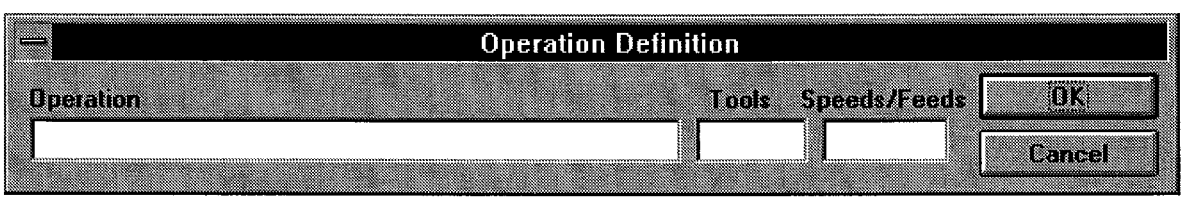
The two large boxes labelled **Special Fixtures** and **Materials** enable the user to enter any component specific details concerning fixtures and materials. These boxes allow the entry of more than one line of text, as does the large box on the right which is available for any extra notes that need to be maintained.

To associate a machine tool resource to a component, the **New Machine** button should be pressed. A dialogue appears which prompts for a machine tool name which is added to the Machine list when the OK button is pressed. Now the details of that tool can be added in the CNC File, Setup and Time boxes. A Machine may have a name or reference number and may use a CNC file for the production of the component. The two time values are for setting up tools and fixtures and for the actual machining duration.

Note: machines such as robots, conveyors and storage devices (which only handle material) should not be entered at this stage if they do not play a part in altering the shape or value of the component. It is possible to produce a component using any shop-floor layout, providing it contains the correct production machines; however the material handling devices vary from layout to layout and hence should not be defined in the process plan of a particular component. These devices should only be added when a route for the component is created with a particular shop-floor layout.

The operations to be carried out on that machine are edited by means of the Operation buttons. These buttons allow the user to create a new operation; modify and delete existing operations and to insert operations into the list.

This New Operation button creates a Operation Definition form.



The Operation Definition Form

This creates a record of the details relating to a single operation to be carried out on a machine tool used in the production of the component. When the OK button is pressed the new operation is added to the Operation List.

The Operation box could contain a textual description of each operation to be undertaken. The Tools required together with the Spindle Speeds and Feed Rates can also be stored in the appropriate boxes.

Any number of machines may be added to the list for a component and any number of operations can be stored for each machine. When a machine name is highlighted in the Machine List box, the operations for that machine are displayed in the Operations List together with the CNC File and the timings.

Process Planner Controls

The File Menu

New – Clears the current process plan and allows a user to start with an empty form after first prompting to save any data.

Open... – Allows the user to select a process plan from the file store and will open the file for editing.

Save – Uses the name in the Part Name edit box to save the process plan for the component with a **.pp** extension. If there is no name one must be entered.

Print... – Sends the entire Process Plan to the specified printer. The machines and all the operations are listed consecutively.

Print Preview... – Displays a draft printout of the process plan.

Print Setup... – Allows the user to choose and configure a printer.

Exit – Quits the Process Planner application after prompting to save any modified data.

The Form

Route Button – Starts the Denford Route Planner application for the current process plan.

Drawing Button – Sends the drawing filename in the **Drawing** edit box to a CAD package specified in the **pp.ini** initialisation file.

The Machine buttons

New Machine Button – Prompts for a machine name and adds it to the machine list.

Modify – Allows the user to modify a machine name.

Insert – Inserts a new machine before the currently selected machine.

Delete – Deletes the currently selected operation.

The Operation buttons

New Operation Button – Displays the Operation Definition form and adds the new operation to the end of the Operation List.

Modify – Opens the Operation Definition box to allow the user to modify the currently selected operation.

Insert – Inserts a new operation before the currently selected operation.

Delete – Deletes the currently selected operation.

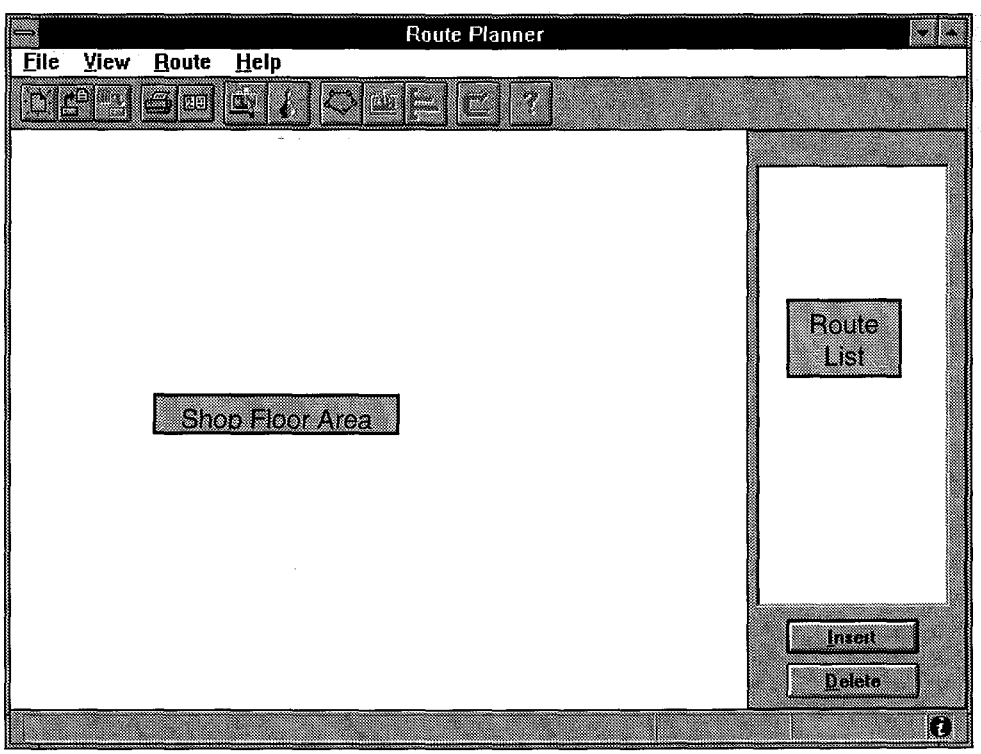
Route Planner

The function of the Route Planner is to fix a component's process plan to a particular shop floor configuration. The shop floor must contain the machine tools necessary to produce the component, together with suitable material handling and storage devices. The user may determine the route for a component around the shop floor, but the Route Planner assists by inferring machine sequences and preventing impossible machine transfers.

Once the route is generated it is time-stamped with the date and time of creation of both the shop floor configuration and the process plan for the component. This prevents any modifications to the component or the shop floor being made without also modifying the route. The route for a component is used in the next step to create a Bill of Processes to control the manufacturing cells.

Usage

The route planner is an application which automatically generates, or allows the user to create, a route for a component around a shop floor. The main interface has a large area which displays the shop floor and a list for the route.

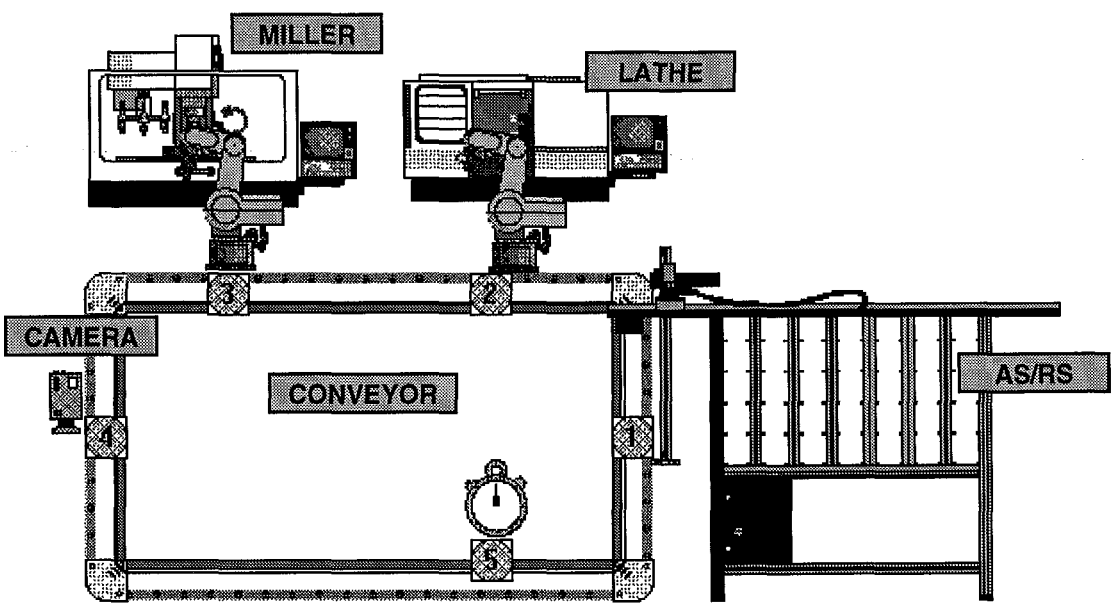


To create a route the user must load a shop floor layout and a process plan by pressing the **Load Process Plan...** and **Load Shop Floor...** menu items or buttons. These buttons present a standard file selection dialogue to let the user choose the two files; the names of the files appear in the status bar at the bottom of the screen.

To quickly generate a minimal route for the component based on the process plan, the **Auto-Route** button may be pressed. This uses the Transportation Network generated by the **Configurator** to create the shortest path from the inventory device to each machine in the process plan and then to the storage device. The complete route is shown in the Route list as a sequence of machines; a graphical representation is obtained by pressing the **Show Route** button. The route is super-imposed onto the shop floor as a graph and the **Show Machines** button is hi-lighted to change the screen back to normal editing mode.

If more flexibility is required, or if the shop floor has devices other than those listed in the process plan which also need to be added to the route, such as vision inspection systems or bar-code readers, there are two alternatives; manually creating a route or modifying an existing route.

To manually create a route, after loading the Shop Floor Layout and the Process Plan, the user may simply press the left mouse button over the machines in the desired sequence. For instance, if the shop floor shown below is used and the component needs to be sent from the AS/RS, to the Lathe, to the Miller, then to the Camera and back to the AS/RS, the user should click the left mouse button over the AS/RS, then on the Lathe, then the Miller, then the Camera and finally on the AS/RS. All transportation devices needed to move the component from machine to machine will automatically be inserted, based upon the transportation network for that shop floor.



When selecting machines in the shop floor window, the Route Planner always checks to see whether intermediate machines are required to allow the

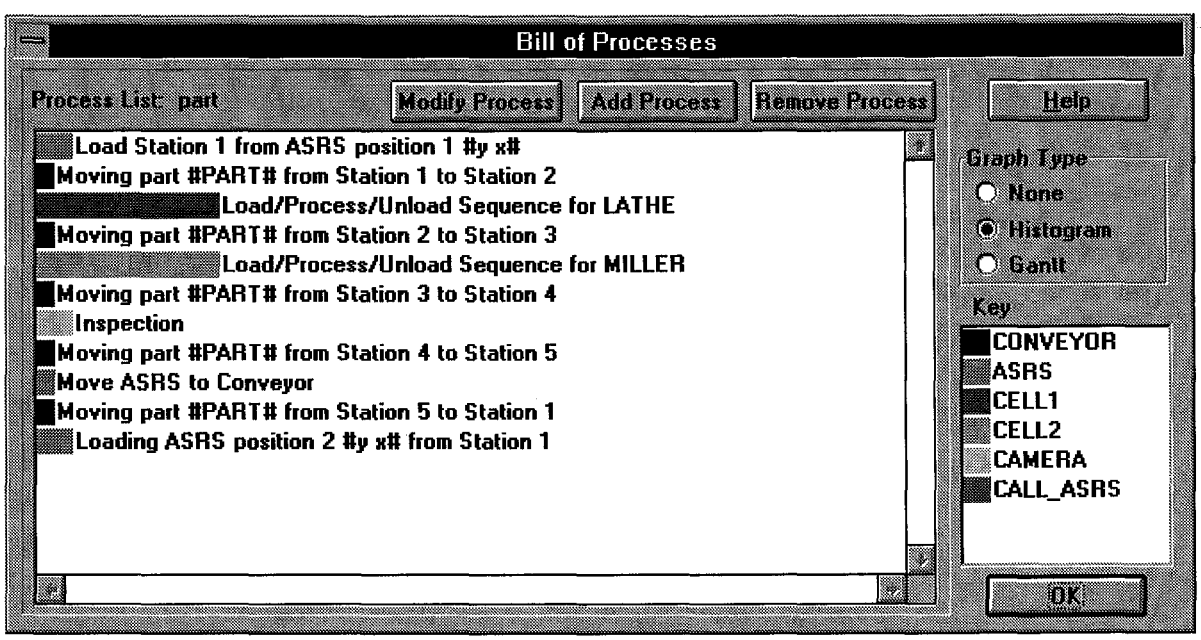
component to move to the last selected machine from the machine before it in the Route List. If transportation devices are required, the Route Planner works out the shortest path between machines and automatically generates and inserts the intermediate route.

Once a route has been generated, it can be modified in two ways. The first method is to select a machine in the Route List and press the **Delete** button. This removes the machine from the route but does not validate the route; the user must ensure that there is still a path from machine to machine.

The second method of modification involves **Insert** button. When a machine is selected in the Route List, pressing the Insert button changes the cursor to an upwards pointing arrow when moved over the Shop Floor window. When a machine in the Shop Floor window is clicked, its name is entered into the Route List above the currently selected machine. Only one machine may be inserted at a time; further selections are appended to the end of the route when further machines are selected. Note, again there is no validity checking for the Insert function; the user must ensure that the Route List shows a valid route.

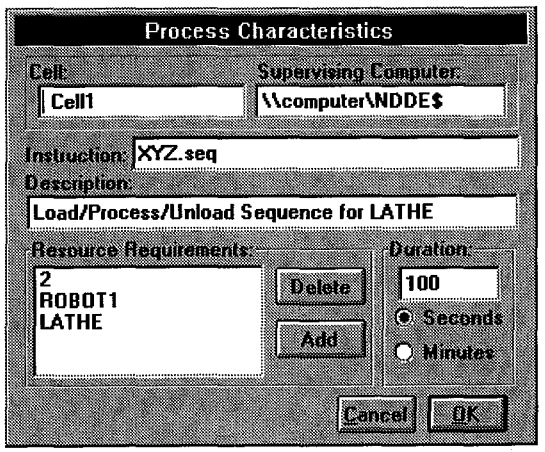
When a valid route has been created, pressing the **Processes**... button creates a sequence of processes to move the component around the shop floor according to the route. The shop floor is defined as a set of cells, each performing a separate function. Each process corresponds to a command sent to a Cell Controller to make it perform a task such as moving a component, unloading a component from stores, or shaping a component.

The processes are displayed as a graph of time against process number, with time along the horizontal axis.



Bill of Processes

The Bill of Processes dialogue shows the graph of processes, colour-coded according to the cell which that process is to control. A key of colours and cells is also given. The graph may be shown as a Gantt chart by pressing the **Gantt** button in the **Graph Type** box. The Gantt chart illustrates more clearly the sequence of processes in time, and is used later in the **Scheduler**.



A process can be modified by selecting it from the list (by clicking the left mouse button over it) and pressing the **Modify Process** button at the top of the dialogue, or by double-clicking the left mouse button on the process itself. When a process is modified, another dialogue appears containing all of the information stored for that process. At the top are two entries containing the cell name and the supervising computer. A process has an **Instruction** which is to be sent to the cell and a **Description** of that instruction. The instruction must be a valid command for the named cell, otherwise an error will occur when the cell comes to execute that instruction. The description is for documentation purposes only, so it may contain anything which is meaningful to the user.

To execute a process, the cell controller must have complete control of a number of machines. Each machine to be used during the execution of a process is listed in the **Resource Requirements** box. This list of resources, held for each process, allows the **Scheduler** to arrange the processes in time without overlapping two processes which require control of the same machine. If two processes overlap causing a machine capacity overload, one process will be rescheduled.

A process has an estimated duration which, in the case of machine tools, is taken from the process plan for the component in question. For other devices the time is a pure estimate and should be altered based on timings taken from the real hardware.

Bill of Processes – Macros

When a Bill of Processes is generated not all the information needed to create machine instructions (or sequence names) is available. Sequence names have to

be input by the user because they are originally created by the user but some machine instructions are automatically generated.

If a machine instruction needs to reference a unique identification number for a component – for example, a conveyor must given a part number for it to carry out a command – a *macro* can be used in the sequence name. A macro is prefixed and suffixed by the '#' character and is substituted for a numeric value by the scheduler when a schedule is created. In the example above, the command to move a pallet containing a 'king' component from station 1 to station 2 on a palletised conveyor is, '**3 king 1 2 #PART#** '. The first digit is a command number which is sent to the conveyor device driver, the remainder is a set of parameters for that command. The first is the component's name, then its location, its destination and then the macro **#PART#**. This is substituted to a pallet identification number by the scheduler, namely a positive number starting at 101.

Another macro is used to describe a component's storage location on an AS/RS. The AS/RS needs an (x, y) position for a pallet when told to either take or replace a pallet from the storage bays. This cannot be known when the Bill of Processes is generated so the '**#y x#** ' macro is used in its place. When the schedule is generated the scheduler queries an inventory database to retrieve the actual y and x positions for a specified component.

Route Planner Controls

The File Menu

New – Resets the Route planner to its initial state. The shop floor, process plan and current route are discarded.

Load – Loads a Route from disk and automatically recalls the corresponding Bill of Processes.

Load Process Plan – Presents the user with a file selection dialogue box to choose a Process Plan.

Load Shop Floor – Presents the user with a file selection dialogue box to choose a Shop Floor Layout.

Save – Saves the Route and Bill of Processes to disk with **.rut** and **.bop** extensions respectively.

Exit – Prompts to save any un-saved data and closes the Route Planner application.

The View Menu

Graph – Graphically displays the route as a set of connected nodes super-imposed on the Shop Floor area.

Machines – Returns the Route planner to standard edit mode after viewing the route as a graph.

Bill of Processes... – If a Bill of Processes has not already been created, creates one and displays it in the Bill of Processes Dialogue (see later).

The Route Menu

Auto Route – Uses the transportation network for chosen Shop Floor to automatically create a route for the component.

Information – Displays the currently loaded shop floor layout, process plan together with their creation times.

The Main Screen

Delete – When an item in the Route List is selected, removes it from the list.

Insert – When an item in the Route List is selected, allows the user to choose a machine from the Shop Floor area to insert above the selected item.

The Bill of Processes

Modify Process – When a process has been selected from the Process List, shows Process Characteristics dialogue box. (see later)

Add Process – with a process selected, allows the user to insert a new process after it. A dialogue appears, asking which cell the process is to command, and the Process Characteristics box is displayed for editing.

Remove Process – Removes the currently selected process from the Process List.

Graph Type – Changes the Process List to either a Gantt chart, a Histogram or a simple colour key of cells and processes.

OK – Updates the Bill of Processes for this component and closes the dialogue.

The Process Characteristics Dialogue

Cell – The name of the cell to be controlled. Note the name can only be edited in the **Configurator**.

Supervising Computer – The PC where this Cell Manager is running. To edit the Supervising Computer, use the **Configurator**.

Instruction – The command to be sent to the Cell Manager to execute this process.

Description – A textual description of the instruction.

Add – Adds another machine to the Resource Requirements list.

Remove – Removes the currently selected machine from the Resource Requirements list.

Duration – The estimated duration of this process, which may be altered and viewed as seconds or minutes by pressing the appropriate button.

Cancel – Disregards any changes to the process and closes the Process Characteristics box.

OK – Updates the selected process and closes the Process Characteristics box.

The Scheduler

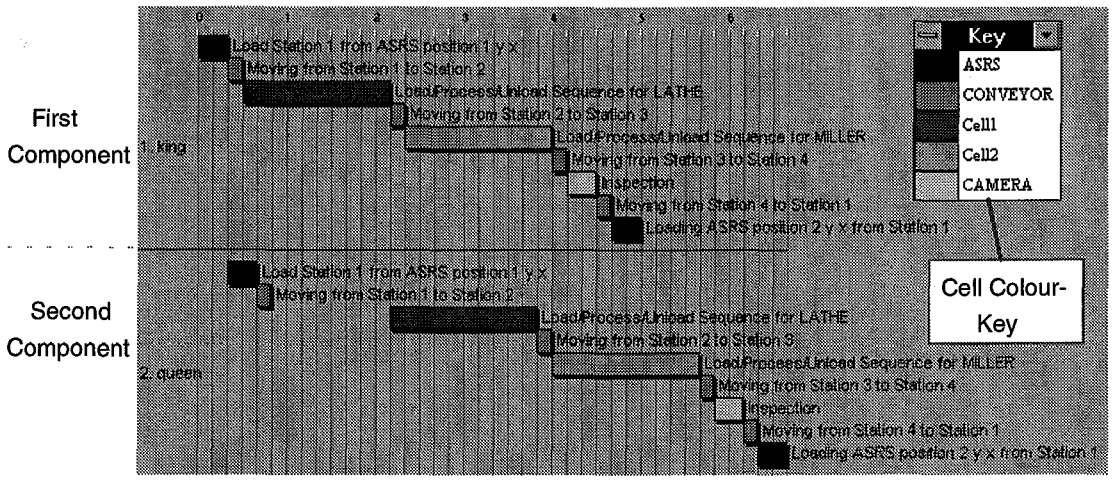
To manufacture a set of components a production schedule has to be developed based upon the routes for the individual components. The Bill of Processes is used for each component and its processes are scheduled in time, ready for production. The scheduling software takes into account the finite capacity of the machine tools and ensures that a feasible schedule is produced. This schedule is displayed as a colour-coded Gantt chart of processes. Each process has an estimated start-time and end-time, based upon its position in the sequence and its duration.

To aid in the analysis of the schedule, the Gantt chart may be viewed in a number of different forms. A chart of machine-utilisation helps to locate production bottle-necks, or the overall cell-utilisation may be viewed. Individual components can also be examined in detail, allowing the user to alter any detail of the schedule. The schedule is then saved to the database ready for real-time execution.

Usage

To create a schedule, select the **New** item from the **File** menu. The Scheduler needs to know which shop floor is to be used for this production schedule so a file selection dialogue is presented. Once the Shop-Floor has been established, the Order Entry dialogue is displayed. This has three columns: component name, quantity and numeric priority.

In order for a process to be scheduled, a valid Bill of Processes must be created with the **Route Planner**. The names of the desired components should be typed, one per line, in the left column with a quantity for each in the central column (the default value for the quantity field is one). If necessary, a priority can be entered in the right column; otherwise the priority takes a default value. When the **OK** button is pressed, a schedule for the components is created. A status dialogue is shown, which indicates the time taken and the number of calculations performed. Pressing the **OK** button on that dialogue shows the schedule.

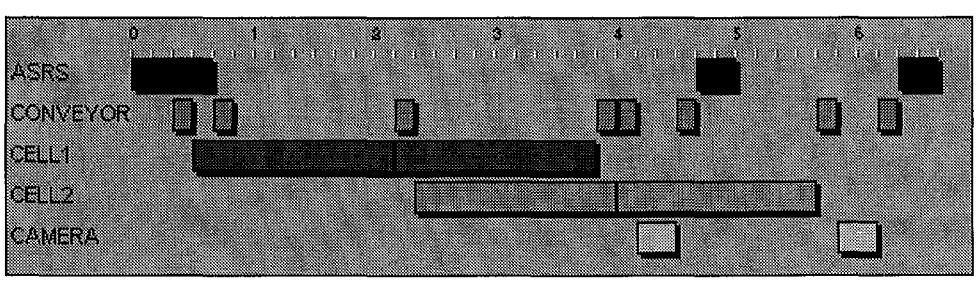


A Schedule for a King and a Queen Component

The schedule is generated from the Bill of Processes for each component in the Order Entry. It is displayed as a staggered Gantt chart of processes, grouped as components, with time along the horizontal axis.

This schedule may be saved to the database or it may be altered and analysed. To save the schedule, the **File** menu's **Save As...** function should be used to give the schedule a name and to store it in the database.

There are a number of ways to view the schedule, accessible through the **View** menu. The **Schedule** item is the default, displaying the Gantt Chart shown above. The **Cells** item re-arranges the schedule to display a Gantt Chart of processes for each cell in the system and the **Machines** item breaks the schedule down further to show when individual machines are busy.



The Cell View

The **Component** item first requests a component number and shows a graph of the individual processes for a component. This screen is similar to the Bill of Processes section of the Route Planner; a Histogram of processes is given, colour coded according to the Cell key. Each process can be altered in two ways: changing its duration by moving the mouse to the right edge and dragging the edge backwards or forwards when the cursor changes to a double-

headed arrow, or clicking on the process bar itself producing the **Process Characteristics** dialogue box. This dialogue can be used in the same way as in the Bill of Processes (except that it has no Add and Remove buttons to modify the Resource Requirements list).

The Component View can also be accessed by clicking the mouse on the component name at the left of the Schedule View.

Scheduler Controls

The File Menu

New – This menu item creates a new schedule. A standard file selection dialogue appears, prompting for a shop floor layout name. The next dialogue is the order entry screen, with three columns: Part, Quantity and Priority. The first column is for a list of part names, one per line (pressing return in the left column starts the next line and hence the next part). The second column is for quantities of the components on the same line as the component. The third column is for a numeric priority between 0 and 9999, 0 being the highest priority. Both the quantity and priority column may be left blank; they take default values of 1 and 9999 respectively. When **OK** is pressed the order is checked for validity before the schedule is produced; (each component should be capable of being manufactured by the named shop floor). The status box is displayed while the schedule is being calculated and can be removed when the schedule is complete by pressing the **OK** button.

Open... – Retrieves a schedule from the database using the standard file selection dialogue and displays it with the default Schedule View.

Save As... – Saves a schedule to the database after allowing the user to name it using a standard file selection dialogue.

Print Preview... – Displays a preview of the printed schedule.

Print... – Sends the current schedule view to be printed.

Print Setup... – Allows the printer to be configured before printing.

Exit – Quits the scheduler.

The Edit Menu

Copy – Makes a copy of the current schedule on to the Clipboard, a storage area which may be shared by several applications, hence data can be transferred between applications.

Paste – Retrieves data from the Clipboard. If there is a valid schedule on the clipboard, it is transferred to the scheduler and displayed. The schedule is then known as an Imported schedule and is displayed with a white border instead of the normal black. If a process in the Imported schedule has been started by the dispatcher before being transferred to the scheduler, it appears as a hatched colour and cannot be modified in any way.

The Schedule Menu

Append – Allows another order to be made and super-imposed on the current schedule which is re-calculated. The current shop floor layout is used so only the Order Entry dialogue is displayed.

The View Menu

Schedule – The default view, showing a colour-coded Gantt chart of processes arranged in groups of components.

Cells – The usage of the individual cells in the system. The cell names are listed on the left and the Gantt chart shows when each cell is busy.

Machines – Similar in layout to the Cell view above, individual machines are listed on the left side.

Component – The processes used in a single component. The duration of a process bar may be altered by dragging its right hand edge with the mouse and any other detail may be altered by clicking the left mouse button on the bar itself.

Order – The component order, as entered by the user may be viewed with this menu item.

Creating the '*layout*' Shop Floor.

Start the **Configurator** from the program manager.

The positions of the machines may be adjusted with the arrow cursor at any time in this design process.

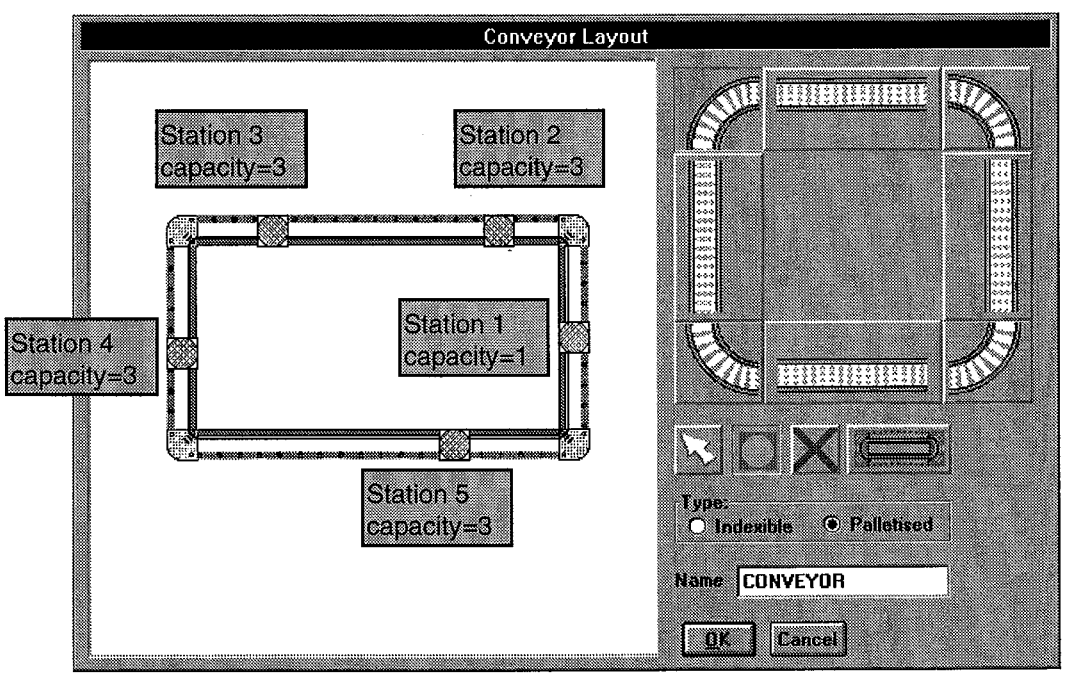
Begin by creating a new Conveyor layout for the system.

Press the Conveyor button to activate the Conveyor Design Utility.

Press the **Palletised** button to create a palletised conveyor.

Move the conveyor sections into place first and then place the five stations on top.

Create a conveyor which looks something like this:



(Note – the first stock conveyor is equivalent to this layout).

Add five stations, starting with the one on the right hand side and working anticlockwise. The capacities of the stations must be as shown, particulary the first which is to be serviced by the AS/RS.

The Conveyor must be a closed loop to function properly. If you make a mistake while building the conveyor use the **Delete Segment** button.

Give the conveyor the name 'CONVEYOR' and press the **OK** button.

The Dispatcher

The function of the dispatcher is to control the set of cells during production. It *executes* the schedule by dispatching work-orders and constantly monitoring the cells for BUSY and IDLE messages. The schedule is updated as events, such as processes starting and finishing, occur on the shop floor. The main screen shows the Gantt chart of processes scheduled in time produced from the scheduling module. A line representing the current time moves across the screen and the schedule is recalculated every second. If a process runs late or finishes early, the other processes in the schedule are altered accordingly. A detailed journal of events is recorded for analysis later. The dispatcher also maintains a journal of status information and sends machine information to the **Simulator**.

Usage

The dispatcher uses a pre-compiled schedule, loaded from the database using the standard file selection dialogue accessible from the **Open** item in the **File** menu. The schedule is displayed in a similar way to the Schedule view of the Scheduler. The components are listed down the vertical axis; their processes are shown as a Gantt Chart with time on the horizontal axis. The processes are all initially white, being coloured to indicate their state once the schedule has started.

The main screen shows a Journal window with the schedule. This window may be moved around the screen and re-sized. During execution of a schedule, the Journal window provides a description of all events occurring in the Dispatcher, including commands being issued and status information being received.

When a schedule is loaded from the database, the Dispatcher immediately attempts to make connections to the cells in the Cell List of the shop floor used to prepare the schedule. Microsoft Windows for Workgroups locates the specified cell computer and replies that it has made a connection to that computer. This connection must be verified by Windows which takes a couple of seconds. If the connection to the Cell Manager has not been made correctly then a Disconnect message is sent and the Dispatcher will be disconnected from the cell. The effect of this procedure is that the Dispatcher first reports that it has made a connection to a cell computer, the connection is then verified and the Dispatcher may be Disconnected if the cell manager cannot be located. Any Disconnection messages are shown in the Journal window.

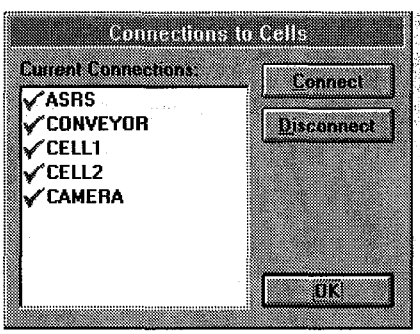
A schedule is executed by pressing the **Start** menu item in the **Execute** menu. If some cells are not connected, a message box asks if execution should

continue. If the user chooses to run the schedule, or if all cells are properly connected, the schedule will be executed.

During execution of a schedule, a vertical bar representing the current time moves from the left edge of the schedule to the right. When this bar reaches the beginning of a process, the process is validated and executed if appropriate. The process will turn green to show that it has started, and the command associated with that process will be dispatched to the appropriate cell controller. If a process is not permitted to start at the current time because of a potential capacity overflow or the late-running of another process, the schedule will be recalculated allowing for this event, with the late running process being given more time or the process causing a capacity overflow being postponed.

When an IDLE message is received from a previously started process, the process has finished and the schedule is recalculated to take this into account.

The execution continues until all components have been completed or until the user presses the **Stop** item in the **Execute** menu. The execution of the schedule is also stopped if an error occurs in a cell, or if a communications breakdown occurs. The error is displayed to the user in the Journal window.



At any time the user can see which cells are connected to the dispatcher with the **Connections** item in the **Service** menu which displays a list of all cells. A tick against the cell name indicates that the Dispatcher is connected; a disconnected cell will be shown with a cross. In this example all the cells are properly connected. A user may select a cell by clicking over it with the left mouse button and forcing a connection or disconnection by pressing the appropriate button.

Other items in the Service menu are **DisconnectAll** and **Reset**. DisconnectAll forces each cell to disconnect communications links with the Dispatcher. This must be carried out before shutting down the dispatcher, otherwise the communication links will be left connected to the Host computer. Any further attempts to connect to the same cell will fail because there is already a broken connection to it. It must be noted that it takes a few seconds to disconnect from all the cells, so the Dispatcher must not shut down until it receives the Disconnection message from each of the Cells.

The **Reset** item closes any previously loaded schedule and resets the time to zero. It returns the Dispatcher to its initial state, ready for a new schedule to be loaded and executed. This action can only be performed when there are no communications links active.

Dispatcher Controls

The File Menu

Open – Loads a schedule from the database and attempts to make connections to the necessary cells.

Print Preview – Shows a preview of the print output.

Print – Sends the current state of the schedule to a printer.

Print Setup – Configures the printer.

Exit – Quits the Dispatcher having confirmed that there are no connections to any cells.

The Edit Menu

Copy – Copies the current schedule, including the state of all processes and the current time, to the clipboard. The schedule can then be pasted into the Scheduler for modification.

Paste – Resets the Dispatcher and loads a new schedule from the clipboard.

The View Menu

Scale – Allows the user to alter the appearance of the schedule by entering horizontal and vertical scaling factors.

Chart – Displays the schedule as a Gantt chart of processes, arranged in component groups.

Cell – Shows the schedule as a Gantt chart of processes for each cell.

The Service Menu

Reset – Resets the Dispatcher to its initial state. The schedule is closed, the timer is returned to zero and all connections are broken.

DisconnectAll – Sends the disconnect message to all cells.

Connections – Shows the list of connections to the user. A cell can be selected and its connection state altered by pressing the **Connect** and **Disconnect** buttons.

The Execute Menu

Start – Executes the schedule from the current time.

Stop – Aborts execution of the schedule.

Appendix A

Configurator

layout.sfl

The shop floor layout.

```
11      No of machines
ASRS        3       1204        256         126       1       0
MILLER      1       1203        39          6         1       0
LATHE       1       1202        155         33        1       0
CONVEYOR    2       1           53          133       0       0
ROBOT1      0       1201        173         80        1       0
ROBOT2      0       1201        63          79        1       0
1           4       1207        234         184       1       0
2           4       1207        197         133       1       0
3           4       1207        96          132       1       0
CALL_ASRS   6       1215        180         204       1       0
4           4       1207        187         238       3       0
```

Machine Name	Machine Type	Machine Code	Xpos	Ypos	Capacity	Extra

Machine Name

A name in upper case containing no spaces, maximum length 14 characters. Numeric names are for stations.

Machine Type

0=robot
1=tool
2=transport
3=inventory
4=station
5=inspection
6=misc.

Machine Code

A code used internally to reference the machine's graphic image.

0=conveyor, the conveyor's name is a file name containing the section details.

1=AGV, as above.

Xpos, Ypos

The position of the machine on the shop floor

Capacity

The component handling capacity of the machine or station. Conveyor=0.

Extra

Not used.

layout.cel

The Cellular structure

```
5                 Number of cells
yo-yo             Computer Name
2 1 1 ASRS        Type, Number of machines m, Machines[1..m], Cell Name
yo-yo             ...
1 1 4 CONVEYOR    ..
yo-yo
0 2 5 3 CELL1
yo-yo
0 2 6 2 CELL2
yo-yo
0 1 10 CALL_ASRS
```

Computer Name

Microsoft's Windows Network Computer Name. (See Control Panel's Network item). This is the name of the computer on which the Cell Manager is to run.

Type

Cell Type.

0=Simple	(one of two machines)
1=Transport	
2=Inventory	
3=Complex	(more than two machines)

Number of Machines

A count of how many devices there are in a cell.

Machines[1..m]

The index number of the machines in a cell according to the .sfl file above. Therefore, CELL1 has 2 machines; 5 (ROBOT1) and 3 (LATHE).

layout.net

The transport network

```
11          Number of machines
1 7         Number of entries m for each machine, Machine List[1..m]
0
0
3 8 9 11
1 3
1 2
1 4
1 5
1 6
0
1 10
```

A list which corresponds to the .sfl file that defines the hierarchy of machine tools. Each machine has a list of machines beneath it in the hierarchy.

Number of entries m

How many devices are below this machine in the hierarchy

Machine List[1..m]

A list of machine indices as stored in the .sfl table.

conveyor.dev

A transport devices' file.

```
Conveyor                File type
100 10                  Conveyor type, number of segments
1105 25 100             Segment ID, xpos, ypos
1105 100 100
1102 175 100
1107 175 25
1101 175 0
1104 100 0
1104 25 0
1103 0 0
1106 0 25
1100 0 100
200 125                 width, height
```

File Type

Conveyor means it's a conveyor's file. AGV means it's an AGV's file.

Conveyor Type

0=Indexible

100=Palletised

Number of Segments

A count of how many segments are to follow in the list.

Segment ID

An internal reference to the segment's graphic

Xpos, Ypos

The Segments position relative to the conveyor's top left corner.

Width, Height

The Conveyor's size.

Process Planner

The fields in this file correspond directly to the fields of the application itself.

part.pp

1	**Number of lines following**
Denford Machine Tools Ltd	**Information field**
part	**part name**
24800-1200	**part code**
C:\CIM\PARTS\FMS-119.DWG	**part drawing**
1	**Number of lines following**
30mm aluminium bar	**Material**
2	**Number of lines following**
LATHE/none	**Fixtures**
MILLER/none	.
2	**Number of machine tools**
LATHE	**First Machine Tool**
lathe.cnc	**Machine program**
0.00	**Setup time**
100	**Process time**
4	**Number of lines following**
Face off	**Description 1**
Rough Cut Profile	**Description 2**
Back cut Profile	**Description 3**
Finish Cut Profile	**Description 4**
T1	**Tooling 1**
T2	**Tooling 2**
T4	**Tooling 3**
T3	**Tooling 4**
2000/300	**Speeds/Feeds 1**
3000/400	**Speeds/Feeds 2**
3000/200	**Speeds/Feeds 3**
3000/400	**Speeds/Feeds 4**
MILLER	**Second Component**
miller.cnc	
0.00	
150	
3	
Cut Crown Detail	
Cut Crown	
Cut Collar	
T1	
T2	
T4	
1000/100	
1000/100	
1000/100	

Route Planner

part.rut

A route for a component about a shop floor

```
part                    part name
781192430 C:\FMS\PARTS\PART.PP          time stamp, process plan
781799380 C:\FMS\LAYOUTS\LAYOUT.SFL     time stamp, shop floor
21      number of elements in the route (n)
1 7 4 8 5 3 5 8 4 9 6 2 6 9 4 11 10 11 4 7 1  elements[1..n]
```

Time Stamp

Numeric representation of the date and time the file was last modified.

Process Plan, Shop Floor

The two files used in the creation of the route.

Number of elements

The count of machine tools used in the route

Elements

The index numbers of the machines as found in the given shop floor file

part.bop

A Bill of Processes for a component

```
part          part name
781192430 C:\FMS\PARTS\PART.PP          time stamp, process plan
781799380 C:\FMS\LAYOUTS\LAYOUT.CEL     time stamp, cell config.
9             number of processes following
0 20 2 1 7 7 0 0 1000000 20 1000000 1 FROM 0 1 y x TO 1
Load Station 1 from ASRS position 1 y x

1 10 0 8 7 20 1000000 30 1000000 3 1 2 part
Moving from Station 1 to Station 2

2 100 3 8 5 3 0 0 30 1000000 130 1000000 XYZ.seq
Load/Process/Unload Sequence for LATHE

1 10 0 9 8 130 1000000 140 1000000 3 2 3 part
Moving from Station 2 to Station 3

3 100 3 9 6 2 0 0 140 1000000 240 1000000 XYZ.seq
Load/Process/Unload Sequence for MILLER

1 10 0 11 9 240 1000000 250 1000000 3 3 4 part
Moving from Station 3 to Station 4

0 10 2 1 7 0 0 250 1000000 260 1000000 1 FROM 1 TO 3
Move ASRS to Conveyor

1 10 0 7 11 260 1000000 270 1000000 3 4 1 part
Moving from Station 4 to Station 1

0 20 2 7 1 0 7 270 1000000 290 1000000 1 FROM 3 TO 0 2 y x
Loading ASRS position 2 y x from Station 1
```

For each process there is a block of data with the following description:

Cell number,	the index found in the given cell file
Process Duration,	the processes' duration in seconds
no of resources r,	number of machine indices following
Resources[1..r],	the machines used in this process from .sfl file
Load Station index,	the index of the station to be loaded, 0 if none
Unload Station index,	the station to be unloaded, 0 if none
Start [a, b] domain,	domain of earliest start to latest finish time
End [a, b] domain,	domain of earliest start to latest finish time
Sequence name	command to be sent to Cell Manager
Description	English description

Scheduler

part3.shd

A schedule for a number of components

```
781799380 C:\FMS\LAYOUTS\LAYOUT.CEL      time stamp, cell filename
3         number of components
part      part name
9         number of processes following
.
.
.
```

The schedule is composed of a copy of all the Bill of Processes' for the components which were ordered with the appropriate time domains for each process. For a description of the data in this file, refer to the Bill of Processes (.bop) above.

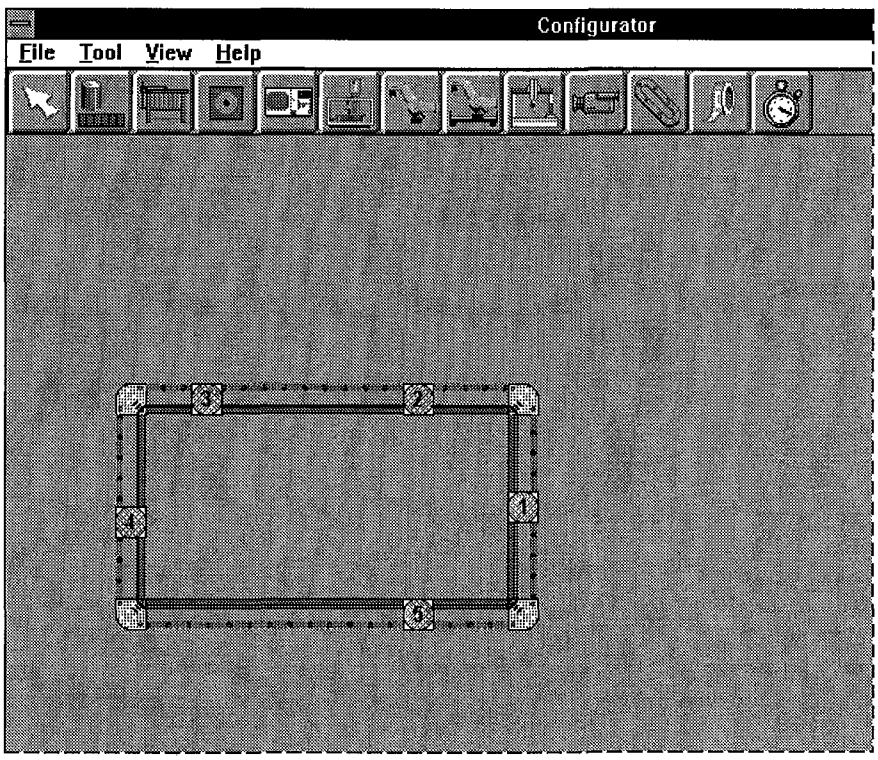
Journal files **dispatch.log** and **sim.log**

Both of these files share a common format. They are used to record the flow of information passing through the **Dispatcher** and the **Simulator** respectively.

```
Date
Time      Message
....
....
Date
Time      Message
....
....
```

The Journal files are appended to each time the applications are run, the first entry being the date and subsequent entries a time and a message sent or received.

Place the conveyor on the shop floor at the left hand side, near the top of the shop floor, by holding the left mouse button down, dragging the convetor's rectangle to the correct place and releasing the mouse button.

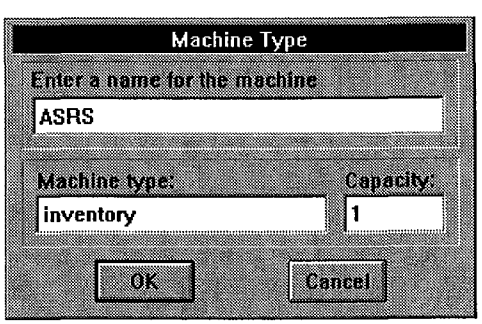


Add an Automated Storage and Retrieval System (AS/RS) to the right of the conveyor

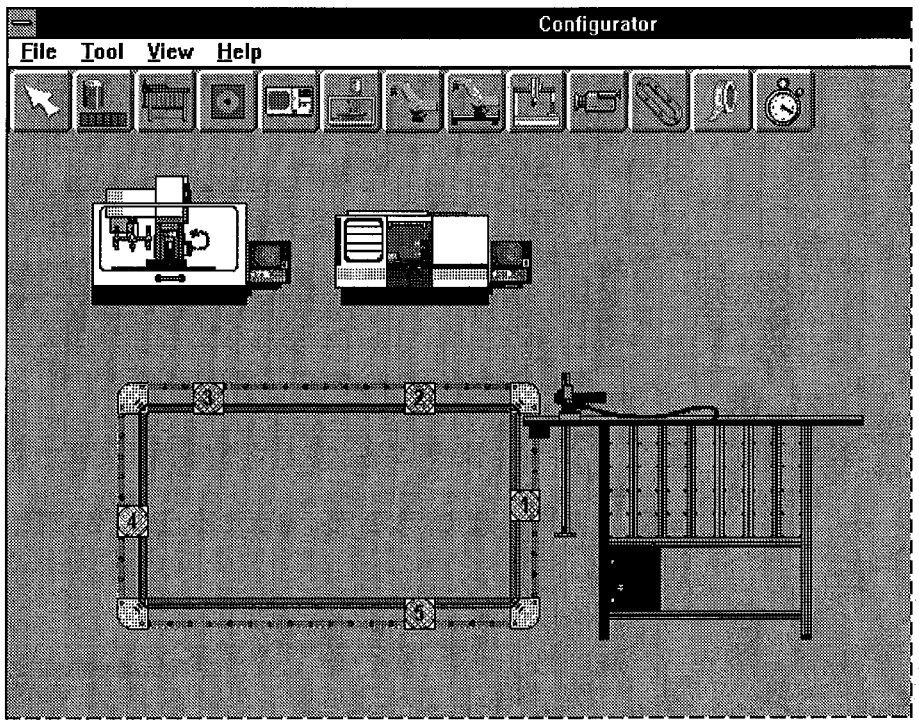
Every shop floor must have some kind of inventory device.

The AS/RS must be close enough to load pallets onto the conveyor.

Call the AS/RS, 'ASRS'; leave its type as Inventory and its capacity as one because it can handle only one component at a time.



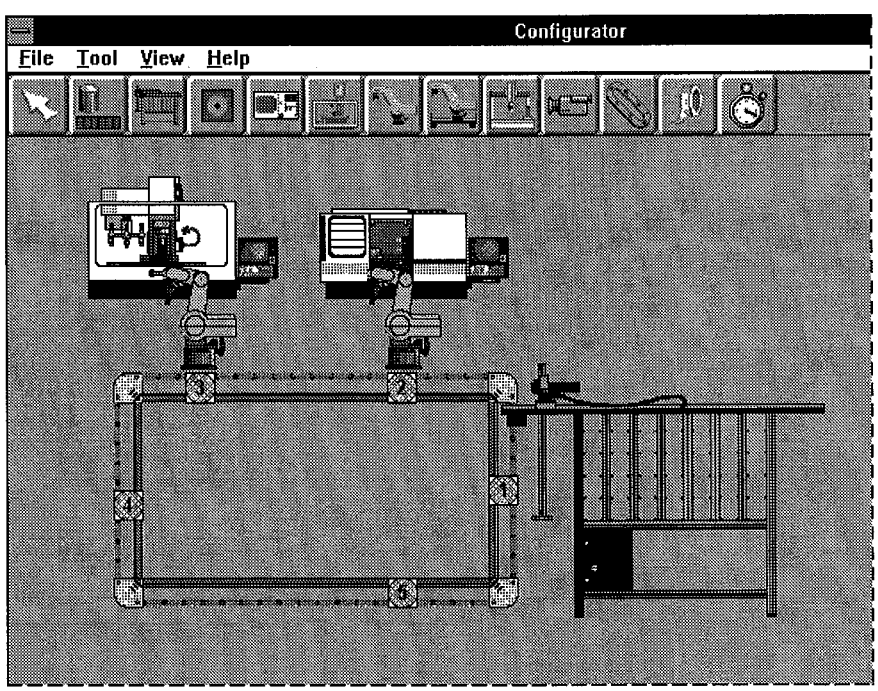
Add a Lathe and a Miller in the positions shown. Leave enough room for a robot between the machine tool and the conveyor.



Call the lathe 'LATHE' and the milling machine 'MILLER'.

Leave their machine types as Tool and their capacities as one.

Add two robots to service the two machine tools.



The robots should be called 'ROBOT1' for the lathe and 'ROBOT2' for the miller.

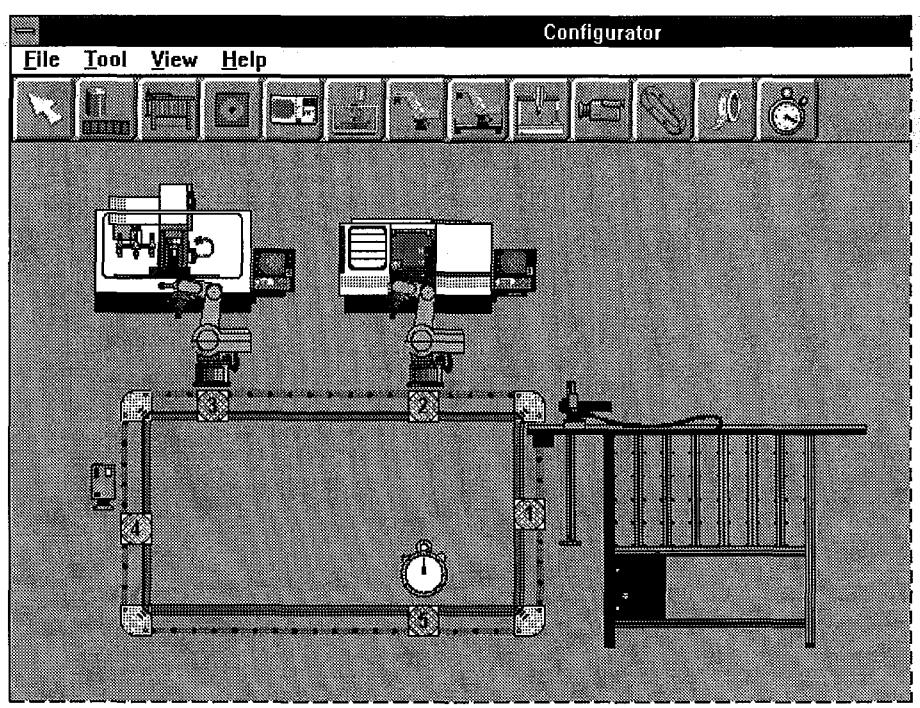
Both robots should have a capacity of one.

You may check the Robot's reach envelope by selecting a robot with the arrow tool and selecting the **Robot|Reach** item from the menu.

The dialogue will list each item the robot can reach and show the envelope as a yellow circle. Every machine within the envelope is highlighted with green. Press the **OK** button to return to the editor.

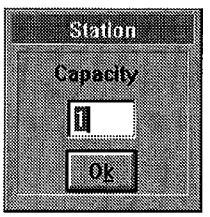
Add a Vision System over station number four. This can be used for component inspection and quality control.

Add a Delay device to the conveyor in the position shown. This device must be present to stop a pallet on the conveyor before it reaches the AS/RS. When a pallet is stopped here a command should be sent to the AS/RS telling it to move into a position to withdraw the pallet from the conveyor. When the AS/RS finishes this command the pallet should be released to move onto the AS/RS. Name the Delay device CALL_ASRS.



The capacity of each station must now be altered to allow for pallet queuing on the conveyor. To alter a station's capacity, double-click the mouse over it, or select it and press the enter key.

The capacity of the first station must be one to prevent the AS/RS from loading a second pallet on top of the first.

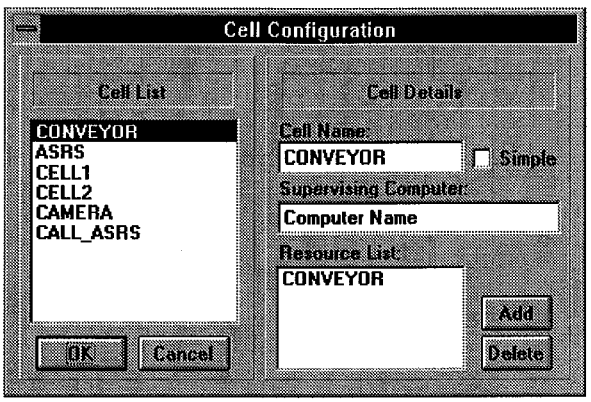


The capacity of the other three stations should be set to three.

Analyse the shop floor you have just created.

Choose **Cell Structure** from the **View** menu

The dialogue box should look like this:



Don't worry if the Cell List shows the cell names in a different order to that shown above.

The cells should contain the following machines. The top row is the Cell Name, following rows show the machines in that cell.

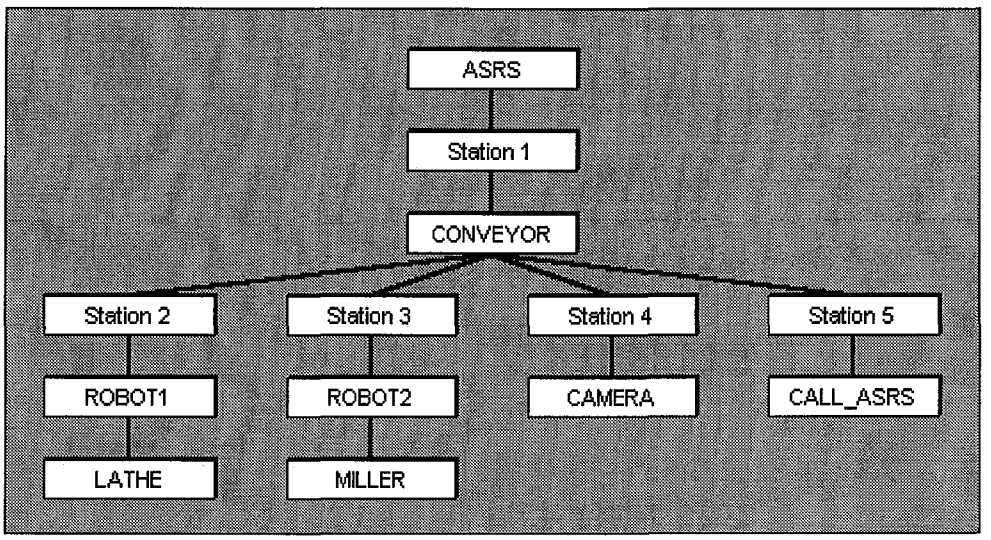
ASRS	CAMERA	CONVEYOR	CELL1	CELL2	CALL_ASRS
ASRS	CAMERA	CONVEYOR	ROBOT1	ROBOT2	CALL_ASRS
			LATHE	MILLER	

If not, see the Trouble shooting section 1.

Close the dialogue by pressing **OK** or **Cancel.**

Choose **Transportation Network** from the **View** Menu.

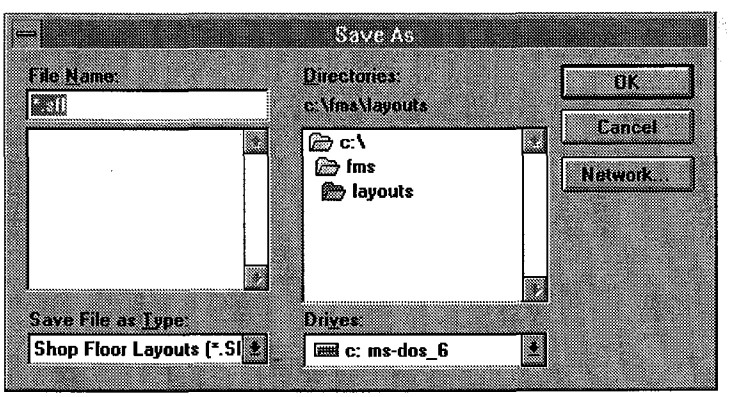
The network should look like this.



If not, see Trouble Shooting 2.

Save the Shop Floor to the database with the **Save As...** item from the **File** menu.

A standard file selection dialogue appears, looking like this:



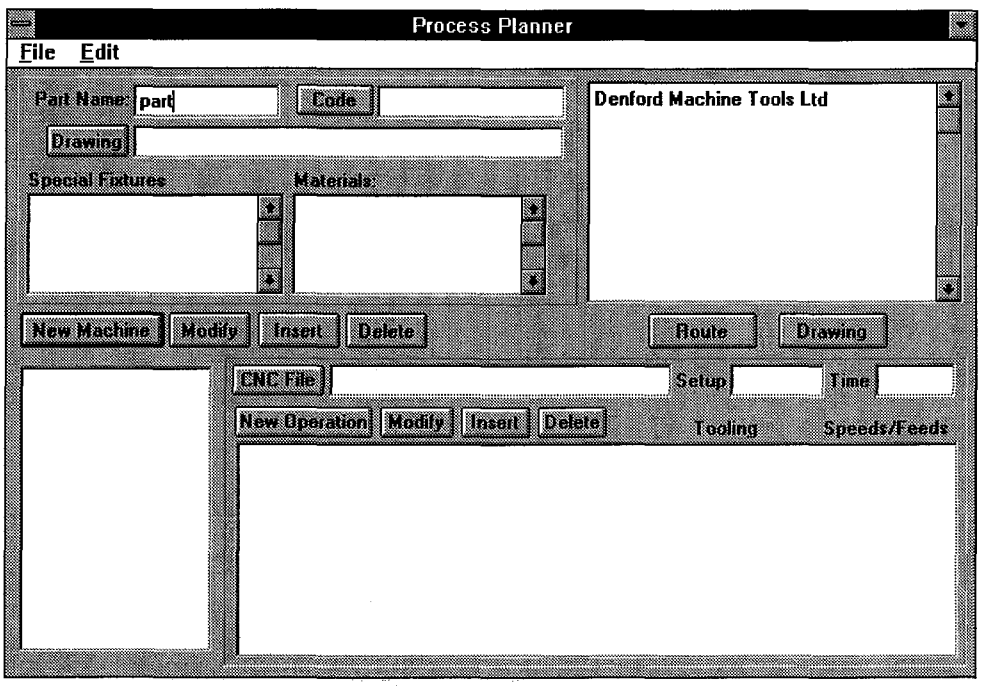
Type the name 'layout' in the **File Name** edit box and press **OK** to save.

Quit the Configurator with **File | Exit** menu item.

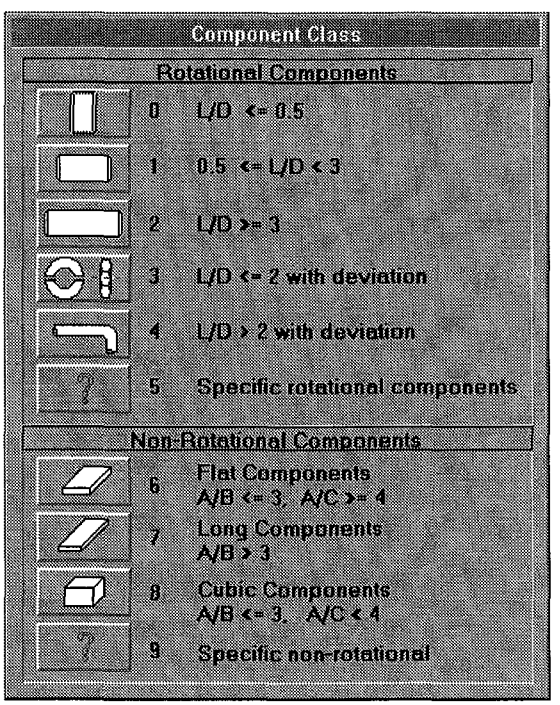
Creating the '*part*' component.

Run the **Process Planner** from the Program Manager.

Type the name 'part' in the **Part Name** edit box.



Press the **Code** button to generate a code for this component.



Generate the code 24800-1200 for the component by pressing the appropriate buttons or keying-in the digits.

Leave the **Drawing** edit box empty for now.

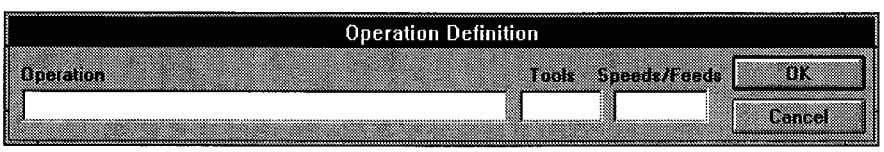
Leave the **Special Fixtures** edit box empty.

Type, '30mm Aluminium bar' in the **Materials** box.

Press the **New Machine** button to enter a new machine tool into the Machining Requirements area. Type the machine tool name 'LATHE' and press return in the New Machine dialog.

Fill in the **CNC File** box or press the button to select a file, for this example, type 'lathe.cnc'. Type 0 in the **Setup** box and 100 in the **Time** box.

Add some operations with the **New Operation** button. A dialog appears with fields for **Operation**, **Tools** and **Speeds/Feeds**.

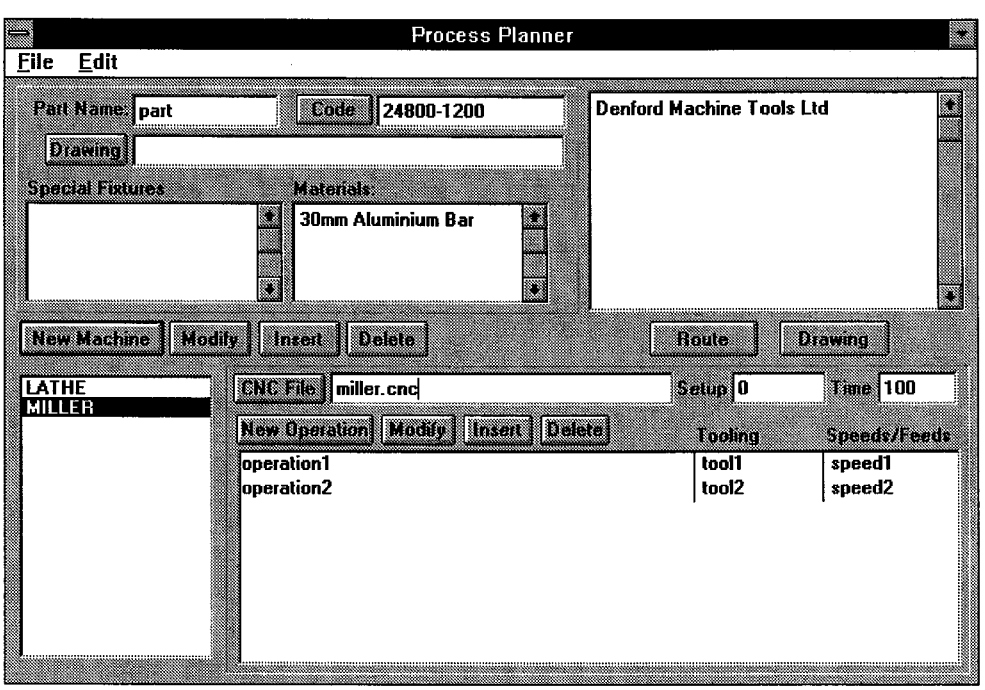


Simply type 'operation1', 'tool1' and 'speed1' and press the **OK** button.

Repeat this to add another operation to the Lathe machine.

Add another machine tool with the New Machine button. This time, call the machine 'MILLER' and add two operations to it.

The Process Plan for the 'part' component should look like this:



Save the Process Plan using the **File|Save** menu item.

Exit the Process Planner using the **File|Exit** command.

Review

We have now accomplished two important things. We have created a shop floor and a component. Note that the component's process plan contains only the machine tools which we placed on the shop floor, i.e. the Lathe and the Miller. These tools actually change the appearance and value of the component. The process plan for a component should not store any material handling or transfer devices because they do not affect the component in any way. The same component may be manufactured on any number of shop floor layouts as long as each shop floor contains machines equivalent to the Lathe and Miller.

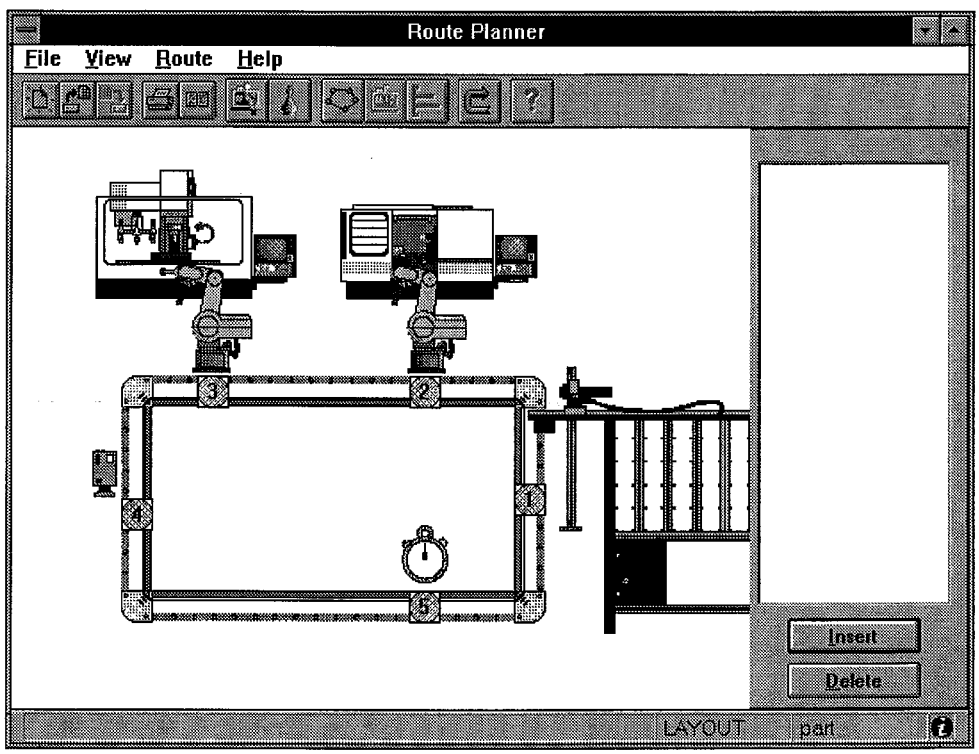
To associate the component 'part' with the 'layout' shop floor a route must be created which defines how the component moves about the shop floor.

Creating a Route for the component 'part' around the 'layout' shop floor.

Run the **Route Planner** from the Program Manger.

Press the **Load Process Plan** button and select the 'part.pp' process plan from the file selection dialogue. Press **OK** to load the file.

Press the **Load Shop Floor** button and select the 'layout.sfl' shop floor from the file selection dialogue. Press **OK** to load the file.

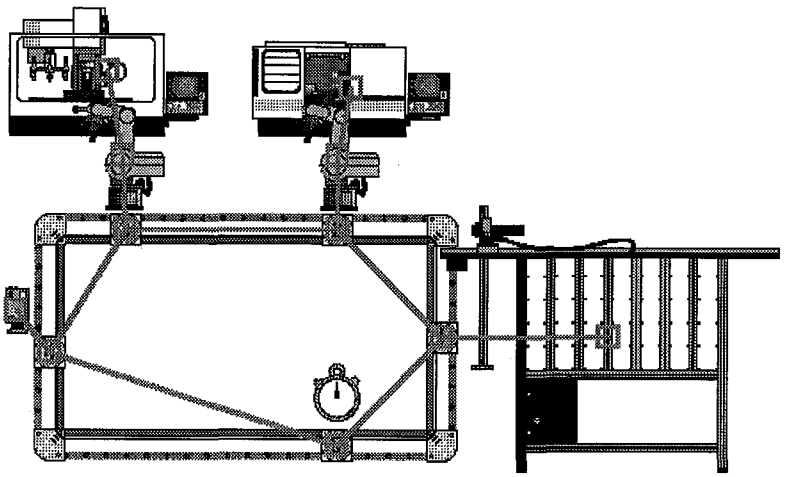


Press the left mouse button on these machines in order:

1. ASRS
2. LATHE
3. MILLER
4. CAMERA
5. CALL_ASRS
6. ASRS

Note how the Route is generated in the **Route** list on the right hand side.

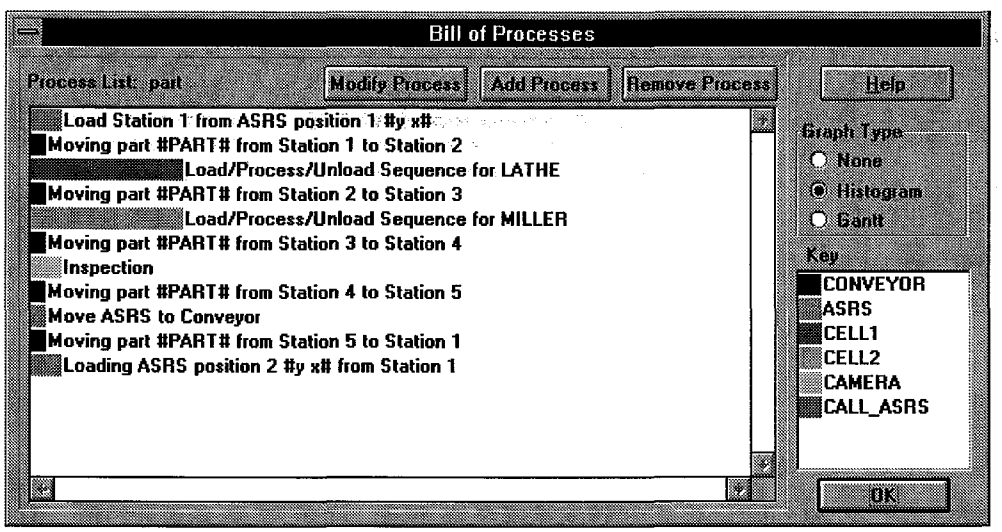
Press the **View|Graph** button to see the route as a directed graph. It should look something like this:



Press the **View|Machines** button to return to the default view.

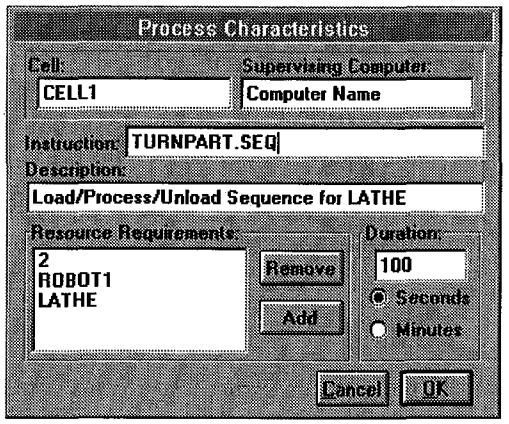
Press the **Processes...** button to view the process plan for this component.

It should look like this:



Modify the '*Load/Process/Unload sequence for LATHE*' process by selecting it with the left mouse button and pressing the **Modify Process** button.

In the Process Characteristics dialogue box change the **Instruction** from XYZ.SEQ to TURNPART.SEQ.



Press the **OK** button to return to the Bill of Processes dialogue.

Modify the '*Load/Process/Unload sequence for MILLER*' process by selecting it with the left mouse button and pressing the **Modify Process** button.

In the Process Characteristics dialogue box change the **Instruction** from XYZ.SEQ to MILLPART.SEQ.

Press the **OK** button to return to the Bill of Processes dialogue.

Press the **OK** button to return to the Route Planner dialogue.

Press the **Save** button to save the Route.

Press the **OK** button to exit the Route Planner.

Creating an Inventory Database

Use Window's Notepad to edit a file called 'invent.dat' in the \cim\bin directory. This file is used by the scheduler to provide coordinates for pallet bays in the AS/RS. The macro **#y x#** created in the Bill of Processes for the 'part' component is replaced by real coordinate values taken from this file.

\cim\bin\invent.dat

5 columns (x)

5 rows (y)

```
part    -    -    -    -
part    -    -    -    -
part    -    -    -    -
part    -    -    -    -
part    -    -    -    -
```

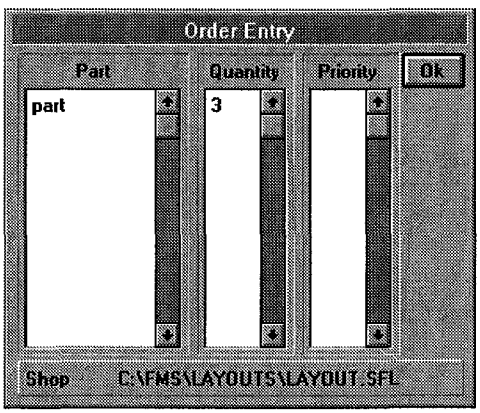
Creating a Schedule to manufacture the 'part' component.

Start the **Scheduler** from the Program Manager.

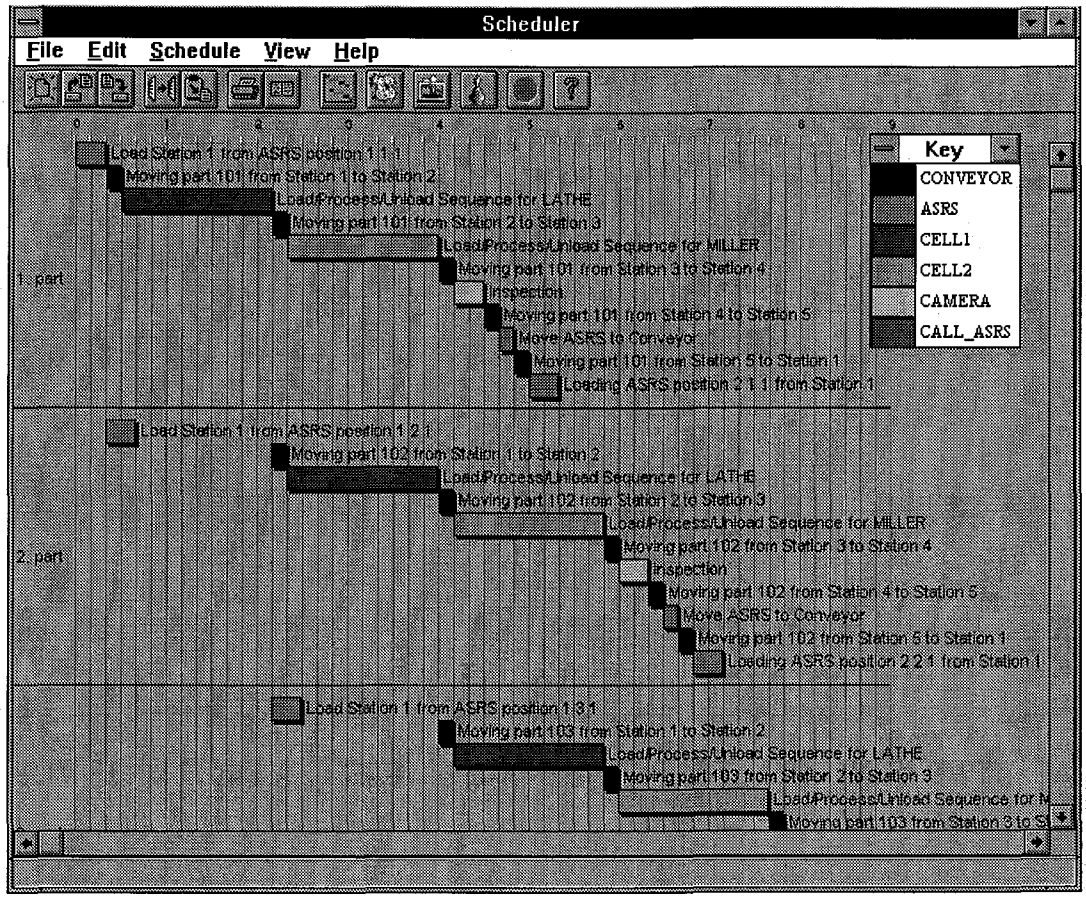
Use the **File|New** menu item to start a new schedule.

Select 'layout.sfl' from the file selection dialogue. This is the shop floor on which the schedule is to be executed. Press the **OK** button to load the shop floor.

Type in 'part' in the **Part** column of the Order Entry dialogue. Type '3' in the **Quantity** column. Leave the **Priority** column blank and press **OK**.

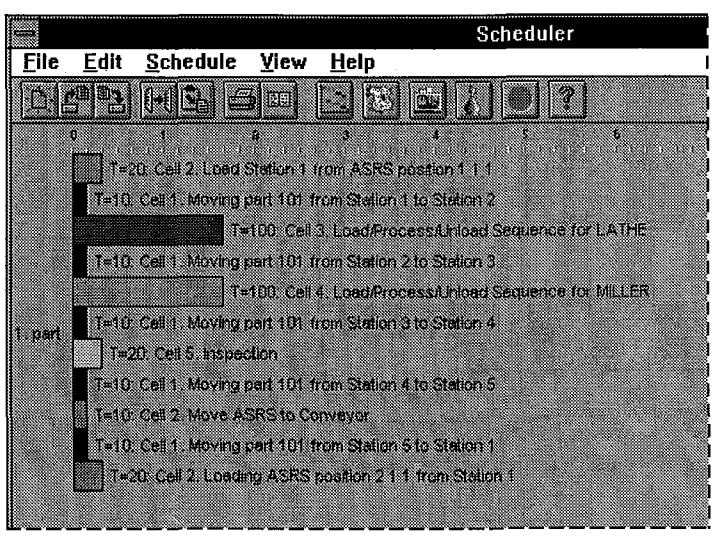


Press the **OK** button on the Schedule Status dialogue to display the new schedule.



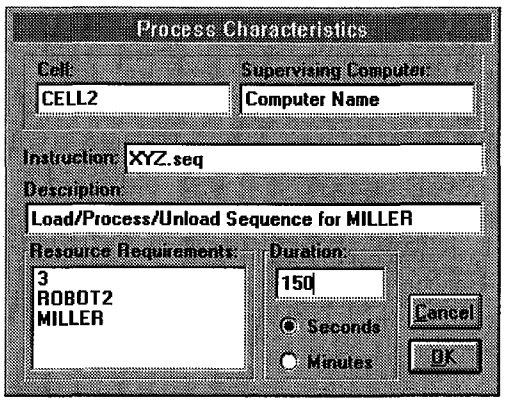
Click the left mouse button on the name of the first component.

This displays the Component View of the Schedule.



Click the left mouse button on the '*Load/Process/Unload sequence for MILLER*' process to show the Process Characteristics dialogue for that process.

Alter the **Duration** to 200 seconds and press **OK**.



Use the **View|Schedule** menu to return to the default schedule view. The scheduler asks if the changes that you made to this component should also be made for all identical components. Answer **Yes**.

The scheduler confirms that all identical components have been changed and displays the new schedule when the **OK** is pressed.

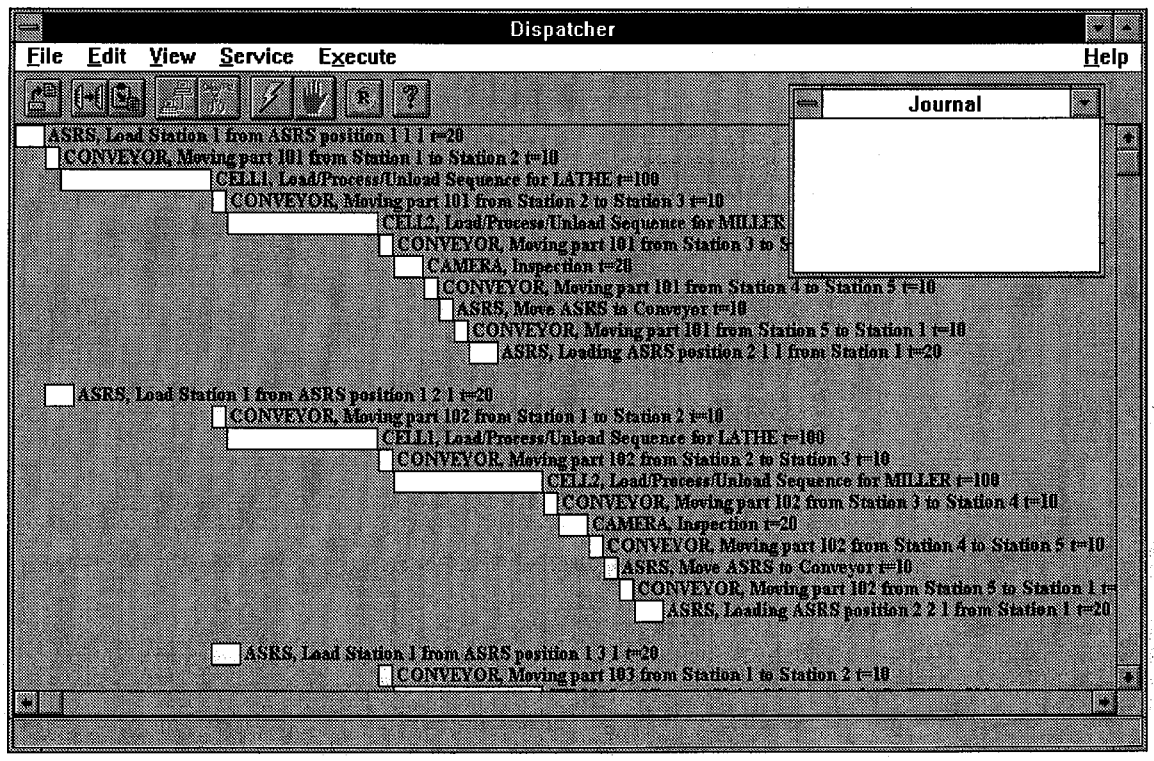
Save the schedule with the **File|Save** command. Give it the name 'part3'.

Exit the Scheduler with **File|Exit**.

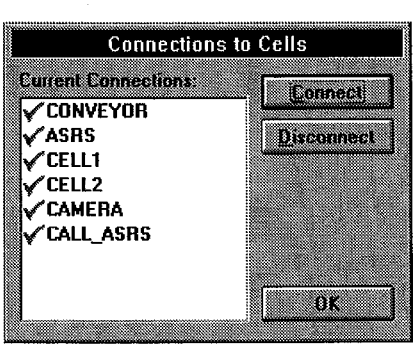
Executing the 'part3' schedule

Run the **Dispatcher** from the Program Manager.

Load the 'part3.shd' schedule with **File|Open**.



Use **View|Connections** to make sure that the Dispatcher is connected to all the cells and device drivers. Refer to section ### for help on starting the cells and device drivers. If the Dispatcher shows a cross next to a cell name it is not connected.



When all the cells are connected and you are ready to begin manufacturing, press the **Start** button or use **Execute|Start** from the menu.

The execution of the schedule can be stopped at any time with the **Execute|Stop** menu item or the **Stop** button.

If a cell returns an Error message or if a cell is disconnected during production then execution will be stopped.

Denford Cell Controller

This manual provides details on the structure and use of the Cellular level of the Denford CIM System. The manual is split into several parts. If you would like to get started with the system as quickly as possible without delving into the manual then turn to the Quick Start chapter. The structure of the cell level of the Denford CIM system is described in the next chapter entitled 'Cell Controller Structure'. The rest of the chapters give details about the actual structure of the cell software. The first such chapter describes the structure of the Cell Manager. The chapters following this describe the Device Drivers, the Interlock Manager and the IO Port Interface.

Cell Controller Structure

Within the Denford CIM system the Cell Control level is situated at the bottom of the hierarchy, below the Host. The Cell Controllers are responsible for communicating with the various manufacturing machines and co-ordinating them in real time. Figure 1 shows the structure of the Cell Controller level of the CIM system and how it links to the Host level. The area within the dashed lines indicates what actually constitutes a Cell Controller. The Cell Controller consists of a Cell Manager and several Device Drivers, one for each machine controlled by the cell.

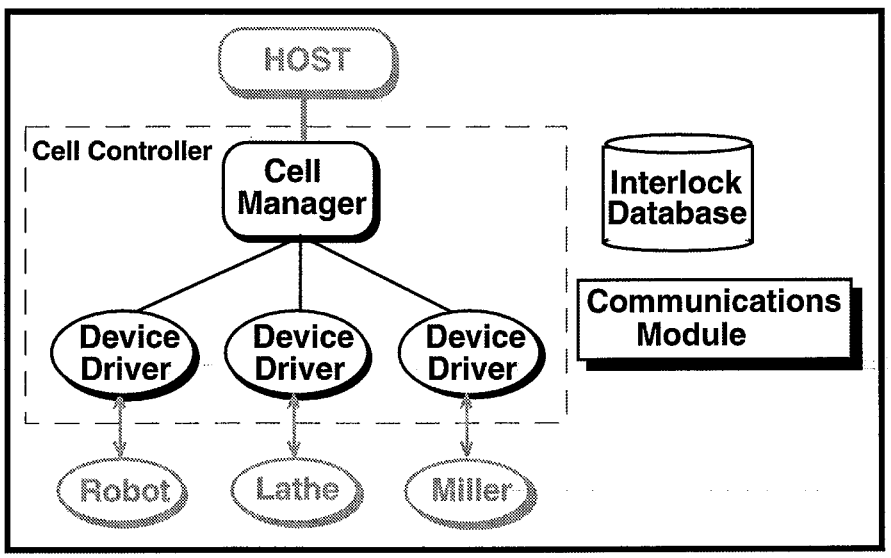
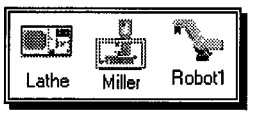


Figure 1 The Cell Level Architecture

In addition there are two modules that are common to all the cell controllers, without which they cannot function. These are the Interlock Database and the Communications Module. At the bottom of the hierarchy are the actual machines themselves, in this example, a robot, a lathe and a miller. There are five main functions performed by the modules which make up the Cell Controller, these are: (i) Sequencing, (ii) Monitoring, (iii) Dispatching, (iv) Interlock Control and (v) Communications.

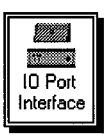
Cell Manager - The overall control of the cell is maintained by the 'Cell Manager'. The Cell Manager performs the machine sequencing (function (i)), it co-ordinates the actions of several machines based on a pre-determined list of instructions. These instructions are stored in a file and are called a 'Cell Sequence'. The Cell Manager sends instructions, at the appropriate point in the sequence, to

the various device drivers under its control and then monitors them for status information. In addition it relays status information about itself and the state of the device drivers onto the Host.



Device Driver - The Monitoring and Dispatching functions ((ii) and (iii)) are implemented by the '*Device Drivers*'. Each device driver is responsible for monitoring and dispatching instructions to one machine only. Dispatching of machine instructions is implemented by downloading programs via RS232 links and or by sending electrical signals via an Input/Output Port. The controllers inside the various manufacturing machines differ widely from machine to machine in both the level of sophistication of controller, from simple relay based control up to an intelligent PLC and the manner in which the controller communicates to other devices. By having a piece of software dedicated to controlling and monitoring a particular machine, its particular communication protocol and control method can be hidden from the rest of the system. Integrating a machine into the system at a later date is only a matter of adding an extra device driver.

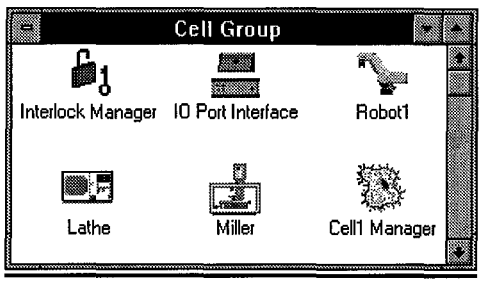
Interlock Database - Synchronisation of machine operations is critical to prevent damaging collisions. For example, consider the case when a robot is taking a part out of a machine tool. Before the robot can enter the machine tool it must know that the door is open and that the machine is not busy machining. In the Denford CIM system '*Interlocks*' are used to safeguard against these situations. These interlocks are implemented as binary semaphores. In this example a machine access interlock is required. Whenever a device requires access to the Machine tool, whether it is the robot attempting to place a part inside it or the machine itself needing access prior to starting some machining, that device attempts to take the interlock. If the interlock is free then the machine takes it and proceeds. The safety is guaranteed because only one machine can at any one time take the interlock. All the interlocks are collected into a database and are administered by a software module called the '*Interlock Manager*'. It is this module that takes care of interlock control (function (v)).



Communications Module - The other module required in the Cell Level of the Denford CIM System is a communications module (function (v)). The communications module is composed of several libraries which cover the various type of

communications required by the system. An Input/Output port or '*IO Port Interface*' library allows the various device drivers to send and receive direct manufacturing standard (0-24V) electrical signals to the machines. These signals are used to send instructions to the machines and to receive status information from them. In addition, an RS232 library enables the drivers to link to some of the machines via an RS232 interface. These RS232 links can be used for downloading large programs such as CNC programs. Finally various network communication libraries are used by all the programs in the Denford CIM system in order to communicate with one another both locally within a PC and over an Ethernet based Local Area Network.

Quick Start Guide

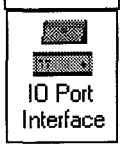


It is very important that all the software is started via the cell group in the program manager. The working directory for each program is assigned within the group and any attempt to start a program from the File Manager will in most cases fail. To activate a program from the program manager simply double click on the programs icon with the mouse or press return after the programs item has been selected.

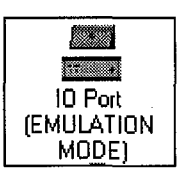
The steps required to start the Cell Software are:

- **Activate the Interlock Manager**

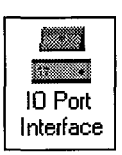
- **Activate the IO Port Interface**



Check that once running the title of the program is 'IO Port Interface'. If it is 'IO Port Emulator' the program is in emulation mode and will not activate any external hardware.



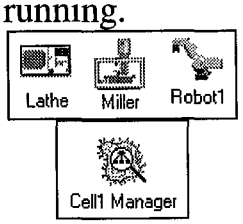
Emulation Mode



Interface Mode

If this is the case close the program and edit the 'iocard.ini' file found in the '\fms\bin' directory. Edit the line beginning with 'Emulation Mode=', change it to read 'Emulation Mode=0'. Before restarting the IO Port Interface make sure no device drivers are running.

- **Activate all the device drivers in any order**



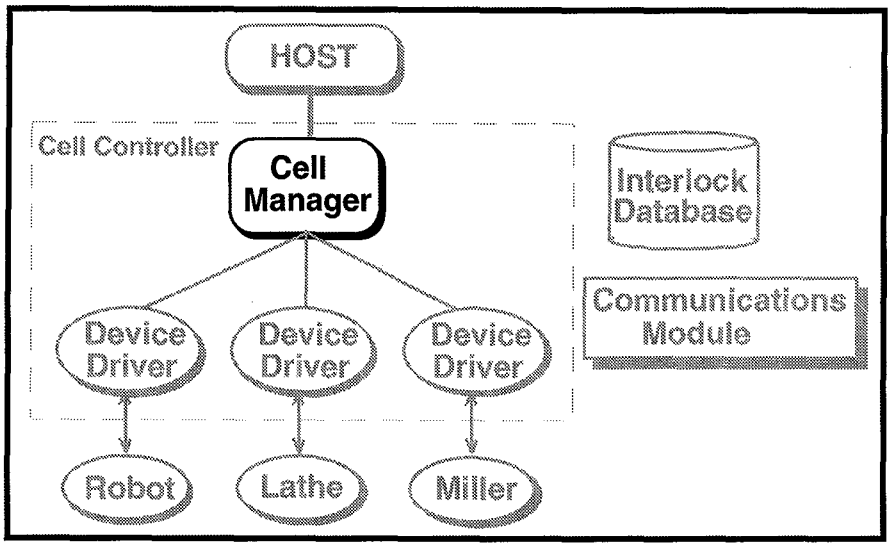
- **Activate the Cell Managers**

- **Check Cell Manager Connections**
  Check that each cell manager is connected to all the device drivers it controls. In the cell machines window of the cell manager any

machine not connected will have a '?' (on a yellow background) displayed in the machines status window. If a machine is not connected then first of all check that its device driver is running, if it is, double click on the machine status window and a dialog box will appear. The dialog box will show the machines status in more detail. Within the dialog box next to the textual description of the machines status is a button. When the cell manager is not connected to the device driver for the machine the status area will display 'NOT CONNECTED' and the button will be labeled '*Connect*'. Simply press the button to connect to the device driver. On connecting with the device driver the machine status area will display something other than 'NOT CONNECTED' and the button will become labeled 'Disconnect'.

- **Run 'Reset' Sequence**
  With each cell manager load its reset sequence and start it manually. This can be done from the main cell manager window which is entitled '*CellX*' where CellX is the name of the cell. Use the 'File->Load' menu option to load the sequence 'reset.seq'. Activate it using the 'Run->Start' menu option.

Cell Manager



Overview

Cell Managers are the software modules found in the Cell control level of the Denford CIM system. Their purpose is to control several machines as an autonomous unit within the CIM. They ease the scheduling burden for the main Denford Scheduler by executing common fixed sequences of machine commands. Each cell manager has a set of commands called Machine Command Sequences which it can be requested to perform. These sequences are stored in files. The user can load a machine command sequence and start it manually with the controls available in the Cell Manager's user interface. The Cell Manager can also receive a request to execute a machine command sequence from another program situated either on the same PC or on another PC through a network (DDEML) link.

The following paragraphs describe an example sequence which put the Cell Manager function into perspective.

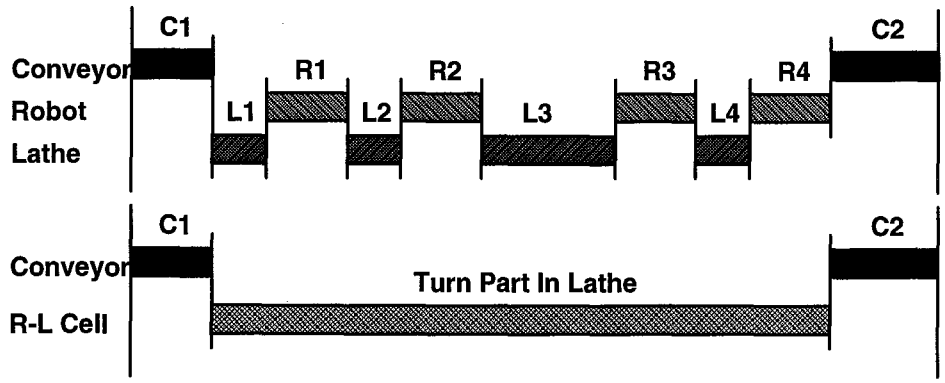


Figure 1 Example Sequence In Gantt Chart Form

Figure 1 shows an Gantt chart of an example portion of a manufacturing sequence. The times are arbitrary and for this example are not significant. In this scenario there is a conveyor, robot and lathe. The conveyor transports a component to the lathe-robot cell. Thereafter the part is placed in the lathe, turned, removed and placed back on the conveyor. Now each bar on the gantt chart shows a machine command being executed. The list of commands is as follows:

C1	Convey the part to the robot-lathe station
C2	Convey the part away from the robot-lathe station
L1	Open the chuck
L2	Close the chuck
L3	Close the guard, turn the part in the lathe then open the guard
L4	Open the chuck
R1	Take the part from the conveyor and place it in the chuck
R2	Open grippers, leave the lathe and park.
R3	Go into the lathe around the part and close the grippers
R4	Place the part back on the conveyor

The top part of the Gantt chart shows the individual commands in detail. The bottom half of the Gantt chart shows how the Cell Manager converts a sequence of commands for the lathe and robot into one command, '*Turn Part In Lathe*' thus removing a large burden from the main host scheduler. When the host scheduler requires the lathe-robot cell to turn a part it will send a '*Turn Part In Lathe*' command to the Cell Manager. At this point the cell manager will load a file which contains the sequence of commands for its device drivers which it subsequently begins to dispatch. Device drivers under the control of the cell manager are

constantly monitored and whenever a command is completed the next one in the sequence is dispatched.

In this example the first command dispatched by the cell manager is L1, 'Open the Chuck' which is sent to the lathe device driver. When the lathe driver receives this command it will execute it and then report a 'BUSY' status while the command is being implemented. When the command is finished the lathe will report an 'IDLE' status. The cell manager monitors the lathe driver and when it detects an 'IDLE' status signal it dispatches command R1, 'Take the part from the conveyor and place it in the chuck' to the robot driver. Again the cell manager monitors the status of the robot driver which becomes BUSY while the command is being executed and returns to an IDLE state once the command has been completed. At this point the next command in the sequence L2, 'Close the chuck' can be dispatched to the Lathe driver. This pattern of monitoring and dispatching continues until the entire sequence of driver commands has been dispatched.

When the request to implement a sequence is first sent to the cell manager a message will be sent to the requester, informing it that the sequence is BUSY i.e., in the process of being implemented. On completing the sequence an IDLE message is sent to the requester so that it knows the sequence has been finished. A requester could be the operator who activated the command manually, or another program on the network (usually the host dispatcher). It is possible within a cell to have several sequences active at one time. For example, a cell containing a robot, lathe and milling machine could have one sequence whereby the robot is loading the miller while simultaneously executing a sequence which involves the lathe turning a part. For this reason it is a sequence state that is indicated as being BUSY or IDLE and not a cell manager's status.

Cell Manager Controls

The cell manager controls are split into several areas as shown in Figure 2.

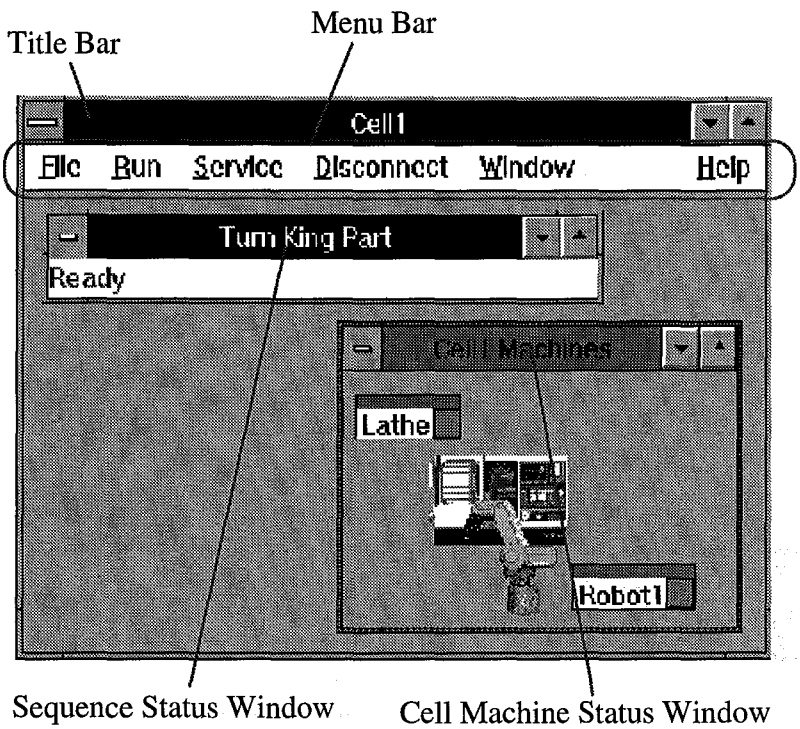


Figure 2 Cell Manager User Interface

Title Bar

The name of the cell manager will always be displayed in the Title Bar. This is the name of the cell manager that is used as a reference to it by all the other programs and modules within the Denford CIM System.

Menu Bar

File Menu

Open This will open a sequence file.

Close This will close the sequence associated with the currently activate cell sequence window. This option is only enabled when the active window in the cell manager user interface is a cell sequence window.

Exit This will make the cell manager terminate.

Service Menu

Disconnect This forces the cell manager to disconnect itself from any remote client program that previously made a DDEML link to it. This will stop the cell manager from receiving any

commands from a remote source. The driver will only be controllable manually.

Run Menu

This menu is associated with cell sequences and is greyed out when the a non-cell sequence window is active.These menu options act upon the currently selected (hi-lighted) sequence.

Start Once loaded a sequence can be started using this menu option.

Stop Once the sequence is running it can subsequently be stopped using this menu option.

Reset This will force the sequence to return to its initial state. When the sequence is subsequently restarted it will start from the beginning of its sequential list of machine commands.

WARNING: Currently there is a problem that causes the cell manager to skip the sending of several machine commands, this occurs when a sequence is stopped using the 'Stop' menu item and then restarted using the 'Start' item.

Service Menu

Disconnect Disconnects the cell manager from its parent, the Host, thus preventing the Host from sending it command requests. At this point the cell manager can only be controlled manually.

Disconnect Menu

All This will disconnect the cell controller from all the machines. Doing this will stop the cell controller from receiving status information from the device drivers or sending commands to them.

Window Menu

Arrange Icons This option will tidy up iconised windows.

Close All This will close all the open sequences.

1,2,3,... This will activate the window whose title is listed next to the number.

Sequence Status Window

This window textually shows the status of a sequence. There is one window per sequence. The window will disappear when the sequence is terminated.

Cell Machines Status Window

This window shows the status of all the machines controlled by the cell manager. A graphical picture of the cell is displayed.

Machine Status Window - For each machine in the cell a small status window is displayed. Each status window has a small caption bar with which it can be positioned in the same way that any ordinary window can. The status window displays the machine name in the left hand side of the window. On the right hand side, a colour coded description of the machine's status is displayed. The colour coding is as follows:

RED with an Asterisk	Indicates that it is 'Busy'
GREEN	Indicates that it is IDLE. Note that there is no charater associated with the idle state.
YELLOW with a question mark	Indicates that the machine's status is not known by the cell manager because it is not connected to it (via a device driver).
YELLOW with an exclamation mark	Indicates that an error has occurred in the associated machine's device driver.

Machine Status Dialog - Double click the left mouse button on a machine status window and a dialog box with a more detailed description of the machine's status will appear. This dialog box is shown in Figure 3. The title bar of the dialog box is labeled with the name of the machine.

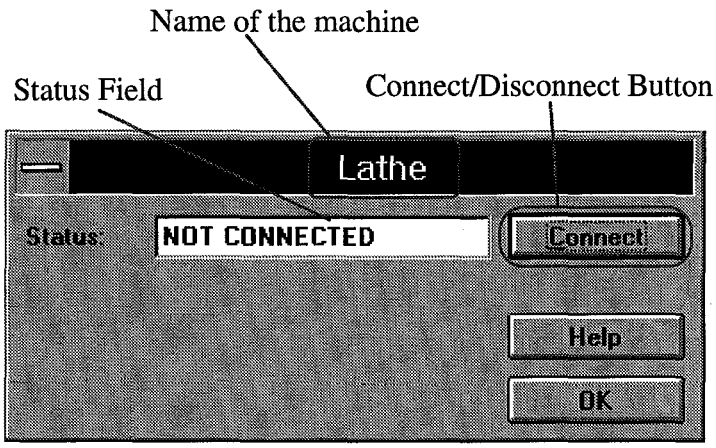


Figure 3 Machine Status Dialog Box

Status Field — At the top of the dialog box is a text box with a more detailed description of the status of the machine.

Connect/Disconnect Button — Next to the status field is a button. Depressing this button toggles the link between the cell controller and the device driver. If the cell controller has a network link (DDEML) to the device driver the button will be labelled '*Disconnect*', depressing it will force the cell controller to break the communication link. When the device driver is disconnected the status field will display 'NOT CONNECTED' and the button will be labelled '*Connect*'. If the cell controller is not connected to the device driver it can not send commands or receive status information.

Hints

The cell controller will attempt to connect to all the device drivers on start up, so it is best to start after them. When an error occurs an error message will be displayed in the sequence status area.

Cell Manager File Formats

Ini File Entries

The following are the set of 'ini' file entries specific to the Cell Manager. An entry in an initialisation file takes the following form:

[section]
entry=value
.
.
.

[GRAPHICS] Section

Bitmap DLL This entry tells the cell manager about the file containing all the graphics information used to display the machines. The user should never need to change this.

[TIMER] Section

Sampling Interval This specifies the time interval in milliseconds that a cell manager will wait before rechecking its status.

[FILES] Section

Data file This specifies the path and name of the cell's data file.

Cell Data File

Associated with each cell manager there is a '*.DAT*' file. This contains details about the structure of the cell that the cell manager is controlling. Figure 4 shows the structure of a cell data file. The file consists of several lines of information. Each line has several fields which are separated by one or more tab characters. The first three lines of the file are a header section and the cell data is defined after these lines.

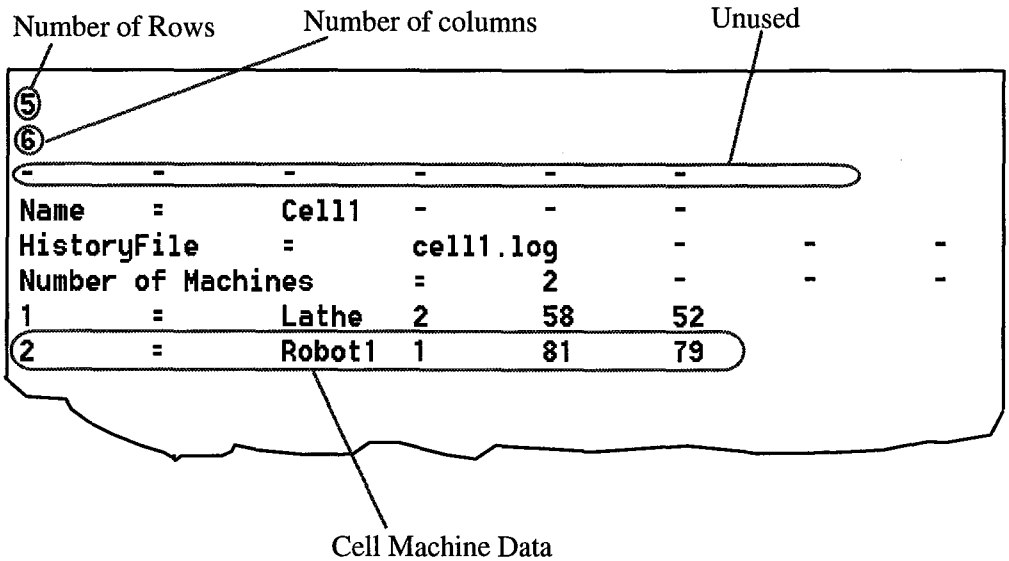


Figure 4 Structure Of A Cell Data File

Header Section

Number Of Rows — This field is the number of data rows in the file, this does not include the first three rows (header section) of the file.

Number Of Columns — This field is the number of data columns in the file.

Unused — Currently the last line in the header section is unused.

Data Definition Section

On the next three lines the data value is placed in the third field.

Cell Name — This is the name of the cell manager within the Denford CIM system. This is the name that will appear on the title bar of the cell manager and the DDEML Service name that all other programs wishing to link to the driver will have to use in order to link to it.

History File — This specifies the name of the file where status history information is logged. The user can change this to whatever they desire but generally it is set to '<Cell Name>.log'.

Number of Machines — This entry specifies the number of machines under the cell manager's control.

Cell Machine Data — The rest of the lines in the file give data about machines used by the cell. The first field in the line is the cell specific identity number for the machine. The second field is an '=' character. In the third field the machines name is specified. The number following the machine name specifies the machine graphics type. The last two

entries in this line detail the position of the machines bitmap. They specify the x and y co-ordinate of the top left hand corner of the machines bitmap respectively.

Sequence File Structure

In this section the structure of a sequence file is described in detail. Figure 5 shows the structure of a sequence file.

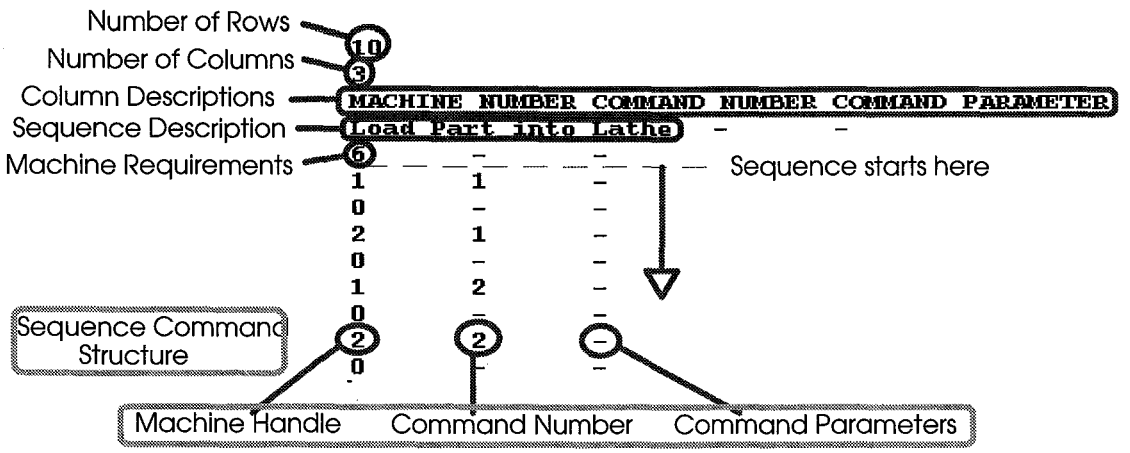


Figure 5 Sequence File Structure

The sequence file consists of several lines of information. Each line has several fields which are separated by one or more tab characters.

Sequence File Line	Description
1 (Number of rows)	In the first row is a figure which describes how many lines there are in the file excluding the first three. In this example there are ten lines following line three.
2 (Number of columns)	The second line describes how many columns there are in the file, this is always three. From the next line onward every line has three fields (separated by tab characters).
3 (Column descriptions)	The third line is a textual comment which gives the headings of the main columns in the file.
4 (Sequence description)	This is a textual description of the function of the sequence file.
5 (Machine requirements)	This line contains one number. This number stores information on which machines are required by the sequence.

6 onwards (Machine command sequence)	In this line onwards the actual machine commands of the sequence and their order are specified

On line 5 the machine requirements of the sequence file are stored as one number. The integer number represents a 16 bit bit-vector. Each bit within the bit vector corresponds to a particular machine as follows:

Bit Number	Decimal Value	Machine
0	1	unused
1	2	Lathe
2	4	Robot1
3	8	Miller
4	16	CMM
5	32	AGV
6	64	Vision System
7	128	Robot2
8	256	Robot3
9	512	ASRS
10	1024	Conveyor

Machine Name Table

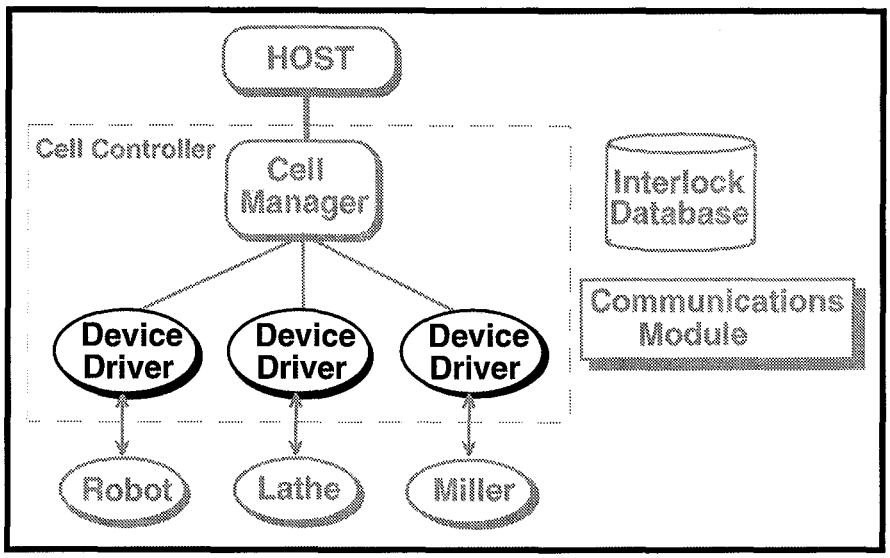
To calculate the machine requirements bit vector, look up the machines required by the sequence in the above table and add together their combined decimal value. The example sequence had a machine requirements bit vector of 6. In binary decimal 6 is 000000000000110 which has bits 1 and 2 set therefore the machines used in this sequence are the Lathe and the Robot. If the sequence required the use of the Miller and Robot2 the machine requirements vector would need bits 3 and 7 set which in Binary is 0000000010001000 and as a decimal this is 8 + 128 = 136. This would be the integer value on line 5 of the sequence file.

The actual command sequence itself consists of several lines, on each line is a machine command. A machine command has three fields, as follows:
Machine Handle - The first field of a machine command is the handle of the machine where the command is to be sent. If this first field is zero then it is not a machine command but a sub-sequence interlock (see below).

Command Number - This is an integer which specifies which command the machine is to execute.
Command Parameters - This field contains extra parameters required by the machine for a specific command. For instance, if the command is a CNC file download command, to be sent to a machine tool, this parameter could consist of the name of the file to download.

When the cell manager starts executing a sequence it sends a 'sequence BUSY' status message to the program or operator which initiated the request. The cell manager executes the sequence in blocks of sub-sequences. Each sub-sequence consists of several machine commands for one or more of the device drivers under the cell manager's control. Within the sequence file the sub-sequences are terminated with a sub-sequence interlock, this is simply a machine command where the machine handle is a zero. All the commands within a sub-sequence are dispatched and the cell manager then waits until they are completed. A sub-sequence can contain several commands for the same machine, in which case it will send the commands one at a time to the device driver in the order in which they are listed in the sequence file waiting for one to be completed before sending the next one. When all the commands in a sub-sequence have been completed the cell manager moves on to the next sub-sequence and implements the dispatch and await completion cycle again and so on until the entire sequence of machine commands have been completed. On completion of the sequence the cell manager returns a 'sequence IDLE' status to the program or operator which requested it.

Device Drivers - Generic



Overview

Device drivers are the lowest level of software in the Cell level architecture of the Denford CIM system. They are situated above the actual machines and interface them to the rest of the CIM system. Device drivers convert the protocol that the machines use into a generic form that the rest of the Denford CIM system will understand. This '*generic protocol*' takes the form

BUSY

When a machine is requested to implement a command it returns a BUSY status until that command has been completed. During this state the machine cannot be requested to perform another command.

IDLE

When a command has been completed an IDLE status is returned. At this point the machine is ready to execute another command.

ERROR

This indicates that an erroneous request has been made to the driver. The driver will require manual intervention to get released from this state.

Device drivers can be controlled manually or requests to the drivers can be made from other programs either on the same PC or another PC on the network via a network (DDEML) link. Each device driver has a set of commands which it can be requested to perform. The commands and their format vary from machine to machine. Most device driver user interfaces will have some kind of list of commands from which the user

can make a selection. The device driver will convert this command into a format that the machine will understand.

There are two types of driver, '*mono-process*' and '*multi-process*'. Virtually all devices fall into the former category, only conveyors fall into the multi-process category. A mono-process driver can only perform one command at a time whereas multi-process drivers can perform several at a time. Thus a robot can only perform one command request at a time whether it is loading a machine, unloading a machine, and so on, it is only physically capable of carrying out one action at a time. Conveyors that can carry multiple numbers of parts or pallets can be implementing different tasks on different parts, pallets or stations. This is relevant to the type of user interface the driver will have. Mono-process drivers have a fairly standard user interface whereas multi-process drivers have unique ones.

Device Driver Controls

Many of the device drivers have a similar look and feel to them particularly the mono-process device drivers. The device driver controls are split into several areas as shown in Figure 2.

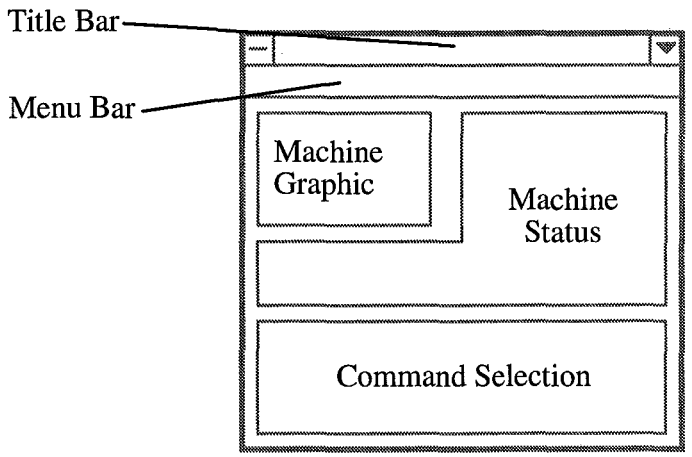


Figure 1 Device Driver Dialog Format

Title Bar

The name of the machine will always be displayed in the Title Bar. This is the name of the machine which is used to reference that driver by all the other programs and modules within the Denford CIM System.

Menu Bar

File Menu

All the device drivers have a File Menu.

Exit This is the only submenu under the file option. This will make the driver terminate.

Service Menu

All device drivers have this menu

Reset This forces the driver to return to its initial state. When a driver goes into an ERROR state this menu option will reset the driver to an IDLE state.

Disconnect This will force the device driver to disconnect itself from any program that has a network link established with it. This will stop the driver from receiving any commands from a remote source. The driver will only be controllable manually.

Machine Status Area

This area contains status information about the device driver.

Control Mode This area specifies whether the driver is under the control of another remote program ('*automatic*' mode), being controlled by the operator ('*manual*' mode) or subject to control by either ('*open manual/automatic*' mode). Currently only the open mode is available to the drivers.

Activity This displays what the machine is currently doing. If the machine is not doing anything then this will read IDLE otherwise the actual type of activity being performed by the machine will be displayed.

Other information displayed in this area is machine dependant. For example, with machine tool device drivers the status of the chuck or vice is displayed in this area.

Command Selection

This contains the means by which the operator can select and activate a command for the machine, under the drivers control, to execute. The list of commands will be displayed in a list box. The operator manually selects a command and depresses the '*Activate*' button. Depressing the activate button will make the driver attempt to execute that command. When the driver is busy executing a command these Command Selection controls are disabled. If the command executes successfully then the '*Activity*' control in the Machine Status Area will display what command the machine is performing.

Device Driver File Formats

Generic Device Driver Ini File Entries

The following are the set of "ini" file entries common to all drivers An entry in an initialisation file must have the following form:

[section]
entry=value
.
.
.

The following sections may contain machine specific entries in addition to the generic ones described here. To find out about these machine specific entries go to the manual section describing that machine.

[IO PORT DATA] Section

This section has the following entry:

Port This entry specifies which port on the Denford Interface Module the machine is connected to. Assign this entry the correct port for the driver by specifying the port name, i.e. '6', '4', '2A', '2B' or '1'. This value can optionally be prefixed with the word 'Port'. For example, to assign port 2A to the driver set the ini-file entry as 'Port = 2A' or 'Port = Port2A'.

[SERVICE] Section

This section only has one entry:

Machine Name This is the name of the driver within the Denford CIM system. This is the name that will appear on the title bar of the driver and the DDEML Service name that all other programs wishing to link to the driver will have to use in order to link to it. It is important that the name used is one of the names available in the Machine Name Table (see Cell Manager - Sequence File Structure section).

[FILES] Section

This has two generic entries:

Status History This specifies the name of the file where status history information is logged. The user can change this to whatever they desire but generally it is set to '<MachineName>.log'.

Commands File This specifies the name of the file in which the commands for the machine are recorded, generally set to '<Machine Name>.cmd'. This should not be changed.

[TIMER] Section

This has only one generic entry.

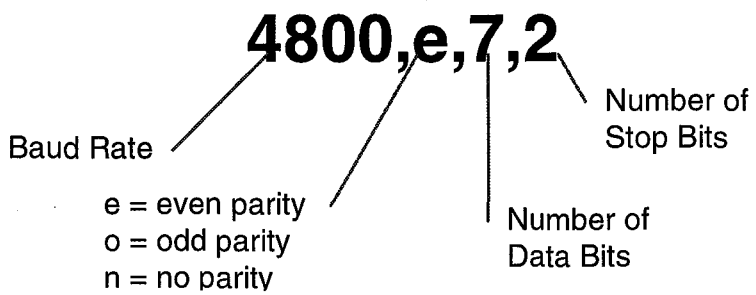
Sampling Interval This specifies the time interval in milliseconds that a driver will wait before rechecking its status.

[COMMS] Section

This section has several entries for specifying RS232 parameters. Only device drivers which have an RS232 link to the machine they are controlling will require this section.

Port This entry specifies the RS232 port on the computer which is used for communicating with the machine by the device driver. The entry takes the form 'COMx' where x is 1,2,3 or 4.

Settings The 'Settings' parameter of the 'COMMS' entry takes the same form as a DOS mode command and it is here where the baud rate, odd or even or no parity, number of data bits and number of stop bits parameters are set.



Flags This entry specifies the type of handshaking to use during RS232 communications. Set to 'H0' for no handshaking, to 'H1' for hardware handshaking and 'H2' for software handshaking using XON/XOFF.

OutBuffer Size Size of the output buffer available during RS232 communications.

InBuffer Size Size of the input buffer available during RS232 communications.

TimeOut Timeout should be a greater value than the longest possible download time.

[COMMAND TIMINGS] Section

This section is special. A device driver does not use any of the entries in this section at all, they are only used by virtual machines. An entry consists of a copy of the text sent to activate a particular command while the value represents the time in seconds to complete that command. Consider the example entries:

3=7

4=1

5 ltest.tap=19

This specifies that a command consisting of the text '3' sent to the virtual machine will take 7 seconds to complete. The command consisting of the text '4' will take 7 seconds to complete while the command '5 ltest.tap' will take 19 seconds to complete.

Format of Command File

The command file contains important information about the machine commands. The command file data generally has the extension '*.cmd*'. The format of a command file is shown in Figure 3. The file consists of several lines of information. Each line has several fields which are separated by one or more tab characters.

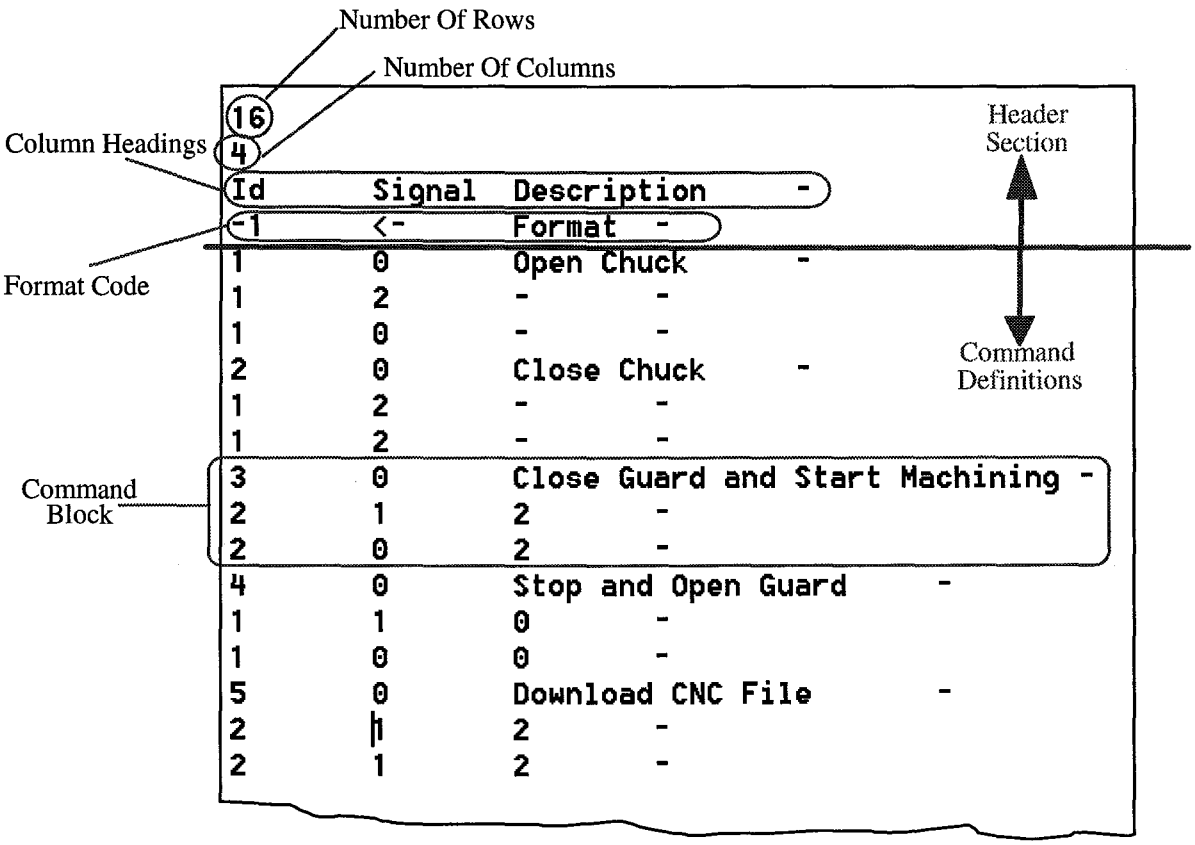


Figure 3 Example Machine Command File (Lathe)

The command file has a header section which is followed by the command definitions.

Header Section

Number Of Rows — This field is the number of data rows in the file, this does not include the first three rows of the file.

Number Of Columns — This field is the number of data columns in the file.

Columns Headings — These fields are the headings of the columns of the first row of each command block.

Format Code — This is a unique integer which tells the CIM software what format the file is in. Currently only a format code of '-1' is supported.

Command Definitions

The command definitions section consists of several '*Command Blocks*'. Each Command Block consists of three lines, see Figure 4.

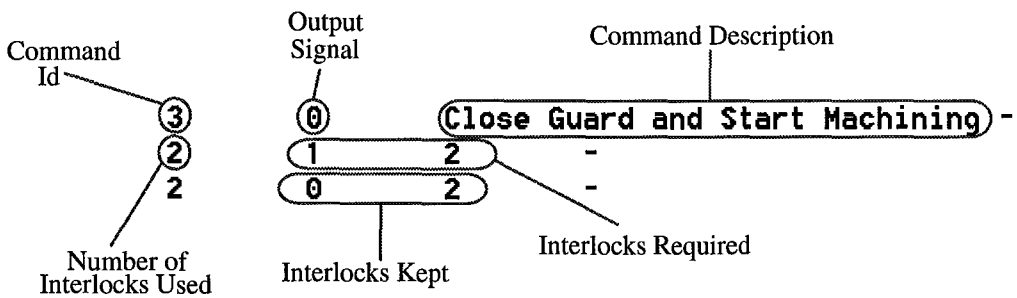


Figure 4 Command Block Details

The first line consists of three fields:

Command Id This number identifies the command

Output Signal This is the decimal value of the binary output sent to the machine's output port when that command is activated. Only the robot device driver uses this field.

Command Description This is a textual description of the command. The user will see this listed in the command selection list box of the device driver's user interface.

The other two lines of the Command Block contain interlock information associated with the command.

Number Of Interlocks Used The first field of the second line and the third line specify how many interlocks are used by the command.

Interlocks Required The rest of the fields on the second line refer to handles of interlocks required by the command before it can commence. The device driver must take all of these interlocks prior to activating the command. If it fails to take any one then it will be unable to activate the command.

Interlocks Kept The rest of the fields on the third line correspond to fields in the same column of the second line but this time denote interlocks which are not relinquished on completion of the command. If a field is set to zero then the corresponding interlock in the second line is relinquished on termination of the command.

In the example Command Block shown in Figure 4 two interlocks are used by the command '*Close Guard and Start Machining*'. These are identified by the Interlock Id numbers '1' and '2'. The interlock with an identity number of '1' is relinquished on termination of the command but '2' is not.

Device Drivers - Machine Specific

Robot Device Driver

The robot device driver is a 'mono-process' driver and as such can only perform one task at a time. The driver is capable of controlling a wide variety of robots as it uses the I/O port only for its communication protocol. When the device driver receives an input of zero, i.e. all the input lines are low, the driver considers the robot to be 'IDLE'. Any non-zero input sends the driver into a 'BUSY' state.

Robot Device Driver Specific Ini File Entries

The following are the set of "ini" file entries specific to the robot device driver.

[IO PORT DATA] Section

See the generic device driver information for details of other entries in this section.

Idle Signal This specifies the value as a decimal number that the robot device driver will read from the I/O port when the robot is idle. By default this is zero. To invert the logic then set the entry to 15 when the robot is connected to a 4 bit port and 63 when it is connected to a 6 bit port.

[INTERLOCKS] Section

Gripper This entry specifies the handle of the robot gripper interlock.

Control This entry specifies the handle of the robot control interlock.

Note that the value of these entries MUST match the values specified in the interlock database file ('semaphrs.dat').

Machine Tool Device Drivers

All lathe and miller device drivers are considered to be members of the machine tool class of device drivers. All machine tools currently distributed within Denford CIM systems are 'mono-process' driver and as such can only perform one task at a time. A machine tool driver is generated for a specific type of machine tool at compile time so any drivers supplied will only be capable of controlling machines they are sent to control.

Machine Tool Device Driver Specific Ini File Entries

The following are the set of "ini" file entries specific to machine tool device drivers.

[IO PORT DATA] Section

Interface Module Port This entry specifies the subset of bits within the IO port used to communicate with the machine (see 'The I/O Port and Denford Interface Module' chapter).

Clamp Bit This can be 0 or 1. Tells the driver which bit in the device drivers IO port corresponds to controlling and monitoring the state of the machine tool's part clamp (chuck or vice).

Machining Cycle Bit This can be 0 or 1 and it specifies to the driver which bit in the device drivers IO port corresponds to the machine cycle status and control. As an input this bit specifies whether the machine tool is in a machining cycle and on the output port is used to send a signal to the machine to start machining.

Machine Idle This can be 0 or 1. This is the value of the 'Machine Cycle Bit' on the input port which indicates that the machine is 'IDLE', i.e. the door is open and the machine tool is not machining.

Clamp Open This can be 0 or 1. This is the value of the 'Clamp Bit' on the input port which indicates that the machine tool's part clamp is open. It is also the value used to force the machine tool's part clamp open when sent to the 'Clamp Bit' on the output port.

As an example consider the following example 'IO PORT DATA' section.

```
[IO PORT DATA]
Interface Module Port = PORT2A
Clamp Bit=1
Machining Cycle Bit=0
Machine Idle=1
Clamp Open =1
```

The port being used by this device is port 2A which uses the IO port bits 12-13. The clamp bit is bit 1 of this address range, i.e. bit 13, while the machining cycle bit is bit 0 of the address range which is bit 12 on the IO port. A signal input of 1 on the machining cycle bit, bit 12, informs the device driver that the machine tool is idle. A signal input of 1 on the clamp open bit, bit 13, informs the device driver that the machine tool's clamp is open.

[INTERLOCKS] Section

Part Clamp This entry specifies the handle of the machine tool's part clamp access interlock.

Machine Access This entry specifies the handle of the machine tool's access interlock.

Note that the value of these entries MUST match the values specified in the interlock database file ('semaphrs.dat').

[FILE EXTENSIONS] Section

CNC Files This tells the driver what the default file extension is for the machine tool's CNC files.

Indexable Conveyor Controls

The indexable conveyor consists of a series of equally spaced cups which move around a track. Unlike the Palletised conveyor the Indexable is an indexing conveyor, the conveyor moves one cup position at a time. A simple mechanical switch detects the presence of the cup.

A indexable conveyor is classified as a multi-process driver, i.e. the device is capable of implementing several commands simultaneously. As a result of this the conveyor requires a special interface. The user interface for the indexable conveyor is shown in Figure 5. The title bar of the driver, as per other drivers, gives its name as the rest of the system sees it. Below the title bar is the menu bar which has the same menu layout as described in the generic device driver interface except that it has an additional 'View' menu. The interface consists of a main window showing the layout of the conveyor. Another window within the main window called the 'Cup Contents Window' shows the contents of the cups.

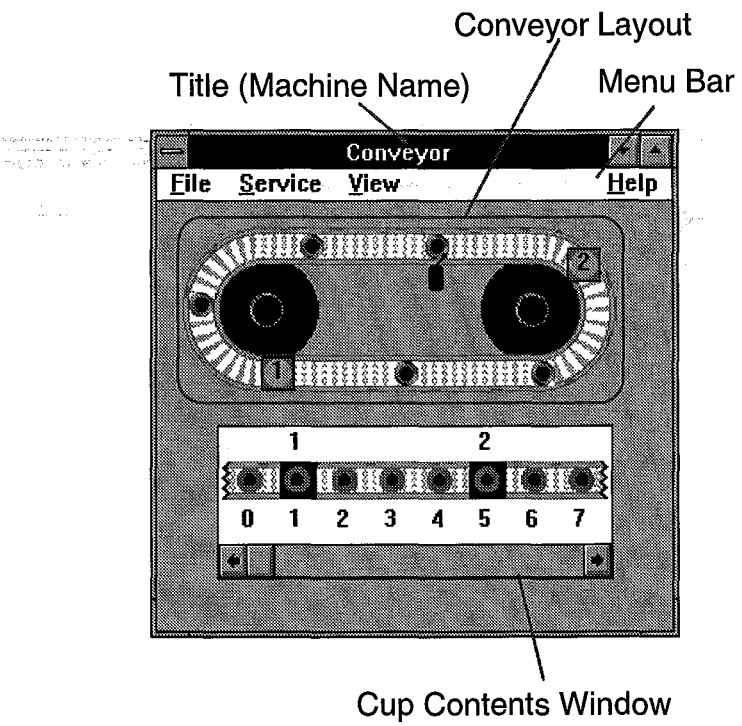


Figure 2 Indexable Conveyor User Interface

Within the main window is another window that displays a section of the conveyor as a continous section, this is the 'Cup Contents Window' as shown in Figure 6. This window graphically displays the conveyor cups and the contents of those cups. The middle portion of the window

displays a section of the conveyor with upto eight cups depicted as a dark grey circle with a black centre. When the cup has a part in it the centre of the cup is coloured red. When the cup as positioned at a station a blue square is drawn around the cup. The 'Cup Contents Window' will change as the conveyor indexes to reflect the new state of the conveyor. The top part of the window shows the identity numbers of the stations. Along the bottom of the window just above the scroll bar, the identity numbers of the cups are displayed. Note that the cup position is relative to the indexing switch. The indexing switch is always at cup zero. The scroll bar along the bottom of the window enables the user to move along the conveyor and view the contents of other cups and stations.

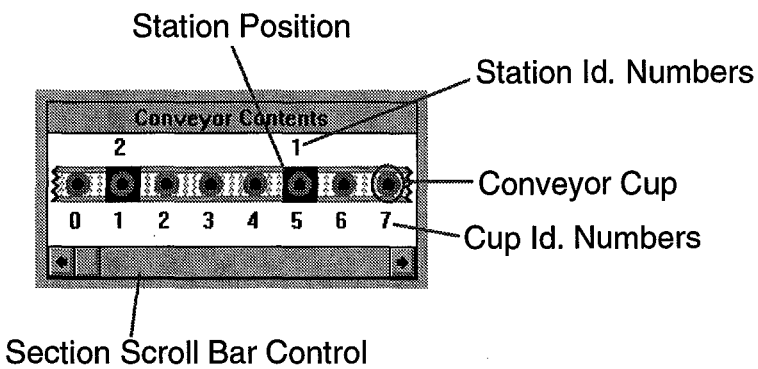


Figure 3 Cup Contents Window

Double click on any of the cups within the cup contents window and a dialog box will pop up which describes the contents of that cup, see Figure 7.

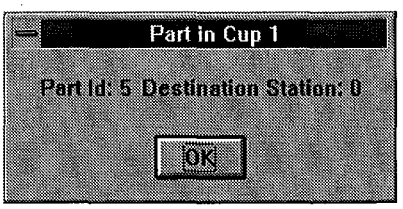


Figure 4 Part In Cup Dialog

Within the dialog box is a description of the part on the chosen conveyor's cup.

Indexable Conveyor Ini File Entries

IO PORT DATA section

Index Signal This is either a 1 or a 0 and specifies whether the signal that indicates that the conveyor is indexing is a high or a low respectively

STATIONS section

In this section the ini file has details about the stations on the conveyor. A station is a cup position where a part can enter or exit from the system.

Number Of Stations — This is the number of stations on the Indexable conveyor.

<Station Identity Number> — For each station there is an entry which is just an identity number for the conveyor. The value associated with this entry specifies where on the conveyor the station is situated. Note that the cup position is relative to the indexing switch. The indexing switch is always at cup zero.

Example ini file STATIONS section

```
[STATIONS]
Number Of Stations=2
1=1
2=5
```

This Indexable Conveyor has two stations, one at cup 1 and one at cup 5.

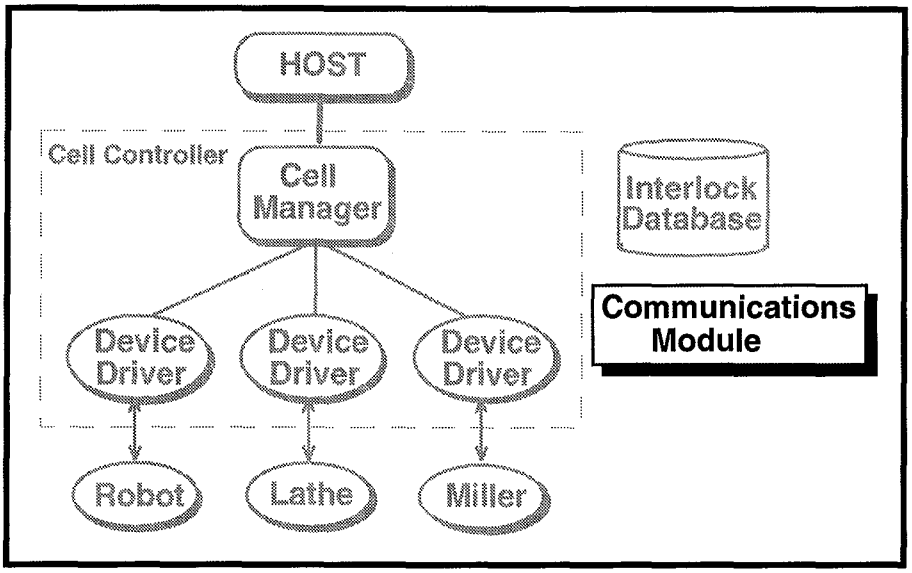
DATA section

Number Of Cups — This specifies the number of cups on the conveyor.

INTERLOCKS section

Index Enable — This entry specifies the handle of the interlock that enables or disables indexing.

The I/O Port Interface



Overview

The I/O Port Interface shows the states of the electrical signals being input to and output from the I/O adapter card mounted inside the PC. This I/O adapter card is linked to the '*Denford Interface Module*' by two 25 way ribbon cables. The labels '6', '4', '2A' etc. correspond to the I/O port sockets on the back of the brown coloured Interface Module. There are two rows of LEDs which show the states of the various input and output lines. These are grouped together under the associated interface module socket. It is important to be aware of the two modes in which the I/O Port Module can operate. These are

Emulation Mode

In this mode the I/O Port module does not connect to the I/O adapter card inside the PC instead it emulates the electrical inputs and outputs in sofware. When it is in this mode the title of the I/O Port Interface reads 'I/O Port (EMULATION MODE)'

Interface Mode

In this mode the LED panel reflects exactly the electrical signals that are being input and output on the IO adapter card. In this mode the title of the program reads 'I/O Port Interface'

Controls

The dialog box and controls for manipulating the I/O Port Interface are shown in Figure 1.

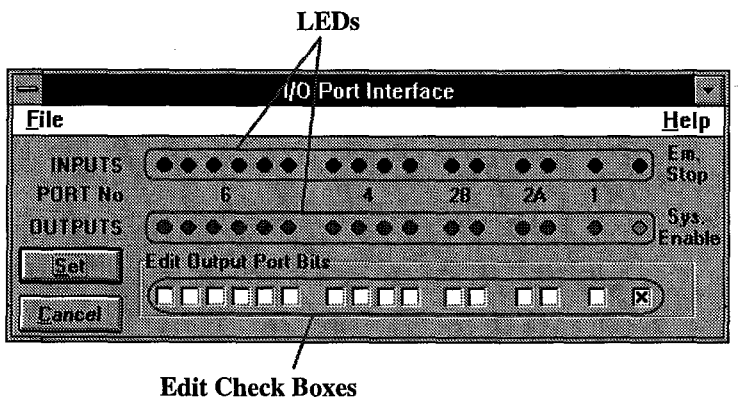


Figure 1 I/O Port Interface Controls

The I/O port interface should never require using. It is useful when introducing new machines to a system to test the I/O port connection to it. The controls consist of a row of Edit Checkboxes, a 'Set' button and a 'cancel' button.

Edit Check Boxes — The check boxes at the bottom of the dialog can be used to edit the settings of the signal lines. When the IO Port interface is in emulation mode the check boxes will edit the emulated INPUT signals. This is so that the user can manually emulate input signals being sent from the FMS machines. In interface mode the check boxes will edit actual OUTPUT signals.

This button will copy the settings of the edit check boxes to the input port (emulation mode) or output port (interface mode).

This button will copy the current settings of the input port (emulation mode) or output port (interface mode) to the edit check boxes.

Ini-File Entries

[TIMER] Section

This has only one generic entry.

Sampling Interval — This specifies the time interval in milliseconds that the IO Port interface program will wait before rechecking the status of the input and output port bits.

[Ports] section

Emulation Mode	The IO Card DLL will run in emulation mode when this is a non-zero value. When this value is set to 1 the IO Card will be controlled by the IO Card DLL.
Input Port	The address of the IO Adaptor cards 16 bit input word within the Cell computers IO address space.
Output Port	The address of the IO Adaptor cards 16 bit output word within the Cell computers IO address space.

[Control] section

Enable	When this is a zero the check boxes within the IO Port interface program will not affect the bits within the IO card
SetEStop	When this is non-zero the Emergency Stop bit on the IO Card will be enabled on loading the IO Card DLL for the first time

The I/O Port and Denford Interface Module

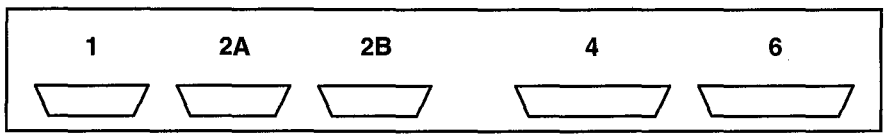


Figure 1 Denford Interface Module I/O Port Connections

On the back of the Denford Interface Module there are several D type female plugs, two 25 way and three 15 way, see Figure 1. Each plug has several input and output lines. The plugs are marked 1, 2A, 2B, 4 and 6, this corresponds to the number of I/O lines going to that port. For example, Port 6 has 6 input lines and 6 output lines. Inside each cell PC there is an I/O adapter card with 16 input bits and 16 output bits. Table 1 shows how the bits of the I/O adapter card map to the following ports. Note that input bit 7 and output bit 7 are reserved for the system Emergency Stop line.

Port	6	4	2B	2A	1	E. Stop
Bits	0-5	8-11	14-15	12-13	9	7

Table 1 Mapping Interface Module Connections to I/O Port Bits

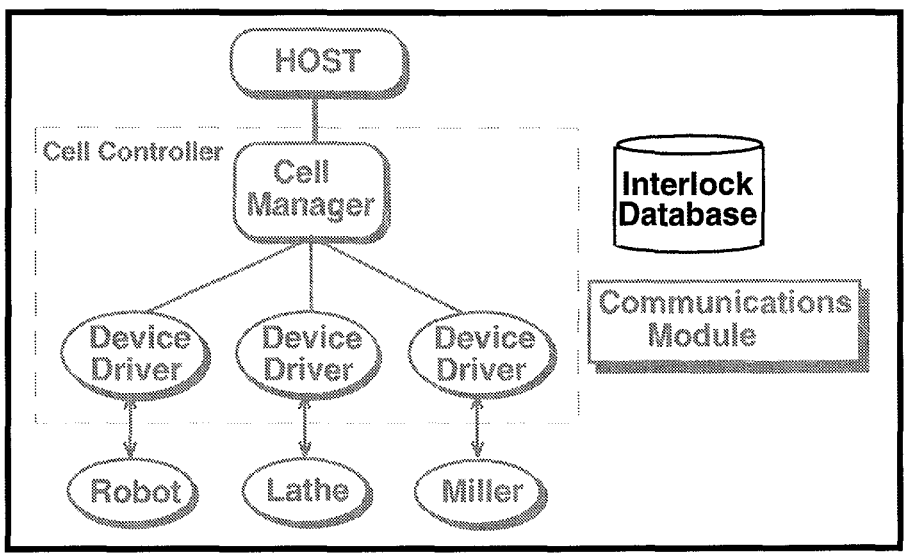
To manage the I/O adapter card there is a special DLL called 'IOCARD.DLL'. Any program that accesses the input or output ports does so through this DLL. To properly initialise the DLL make sure that the IO Port Interface program is executed before any of the device drivers. This DLL will only be unloaded when all the drivers and the I/O port interface program are terminated so any changes to the ini file entries for the I/O card DLL will have no effect until this condition is met.

To assign a device driver to a particular port look inside the driver's ini file. There will be a section called '[IO PORT DATA]'. Under this section are the entries 'LSB' and 'MSB' which are abbreviations for Least Significant Bit and Most Significant Bit respectively. Together these entries indicate the bit range of the inputs and outputs that a driver will use. So a driver using Port 6 will have the entries 'LSB=0' and 'MSB=5'

to indicate that it uses the bit range 0 to 6 inclusively. A driver using Port 2A will have entries 'LSB=12' and 'MSB=13', to change it to use Port 2B change these entries to 'LSB=14' and 'MSB=15'.

In the latest version of the Denford Software there is an alternative easier method to assign a port to a driver. Under the section '[IO Port Data]' is an entry 'Port'. Assign this entry the correct port for the driver by specifying the port name, i.e. '6', '4', '2A', '2B' or '1'. This value can optionally be prefixed with the word 'Port'. For example, to assign port 2A to the driver set the ini-file entry as 'Port = 2A' or 'Port = Port2A'.

The Interlock Manager



Overview

The interlock manager program allows access to the underlying interlocks upon which the machine control system is built.

Controls

The dialog box and controls for manipulating the interlock manager are shown in Figure 1.

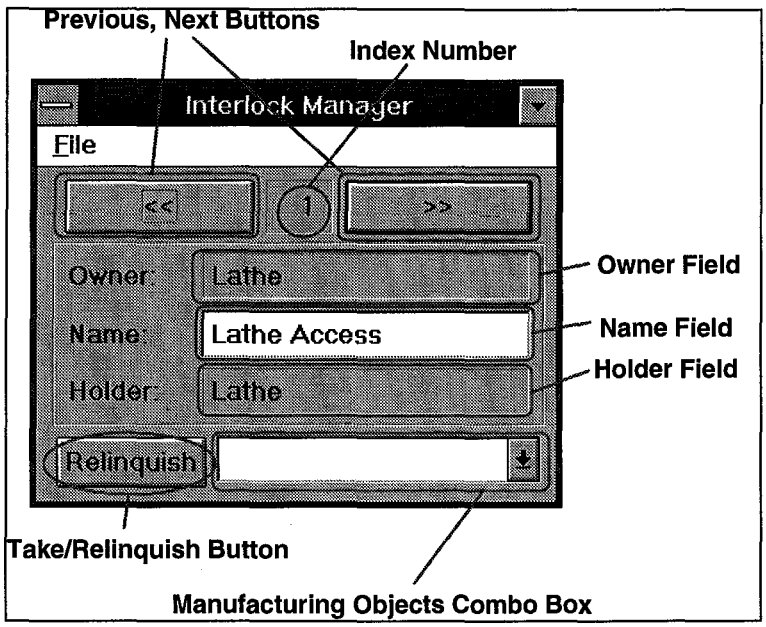


Figure1 Interlock Manager Controls

The main area of the dialog shows the details of a particular interlock. For each interlock three fields are shown

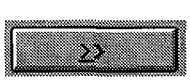
Owner The Owner field identifies the manufacturing object which owns the interlock.

Name A Name field gives a short textual description of the function of the interlock.

Holder The Holder field lists the current manufacturing object which has control of the interlock. Certain actions require as a prerequisite, control of particular interlocks before their implementation can be granted.

At the top of the dialog are two buttons for traversing the interlock database

This button is used to change the interface to show the details of the interlock preceding the current one



This button moves forwards to the next interlock.

Index Number Between the buttons marked '<< and '>> is an index number unique for each interlock.

At the bottom of the interlock manager are two controls that allow the user to manually override the current Holder of the interlock.

Take/Relinquish - If the interlock has an entry in the holder field then this button will have the caption 'Relinquish' inside it. Depressing the button at this point will cause the current holder of the interlock to relinquish it. If the button has no Holder then the button caption will read 'Take'. Depressing the button at this point will cause the currently selected entry in the Manufacturing Objects Combo Box to take control of the interlock

When a run is aborted the interlocks may need to be reset. The best way to do this is to manually operate the device drivers to move them to their start up positions. Manually resetting the interlocks via the Interlock Manager should only be done when a machine is moved out of its current state by manual machine operation. For example if a robot begins inside a machine tool and is then moved, using its teach box, out of the machine back to a park position, then the device driver will not be aware of its change in position. The device driver will need to be reset and the interlocks relating to the robot's previous position manually adjusted to reflect its new state. To change an interlock holder first use the Relinquish button to force the current controller to release it, then choose a manufacturing object from the pull down list next to the Take/Relinquish button, finally press the 'Take' button to force the interlock Holder to become the selected machine.

Interlock Manager File Formats

The interlock manager reads in interlock data from a file called 'semaphrs.dat'. This file is stored in the '\fms\bin' directory. The format of a this file is shown in Figure 2. The file consists of several lines of information. Each line has several fields which are separated by one or more tab characters. The file has a header section which is followed by the interlock definitions.

Header Section

Number Of Rows This field is the number of data rows in the file, this does not include the first three rows of the file.

Number Of Columns This field is the number of data columns in the file.

Columns Headings These fields are the headings of the columns of the first row of each command block.

Interlock Definitions

Each row of the file in this section defines an interlock.

Interlock Handle The first field identifies the handle of the interlock.

Interlock Name The second field denotes the name or functional description of the interlock.

Owner In the third field is an integer which identifies the manufacturing object which owns the interlock.

Holder This field contains an integer identifying the manufacturing object which initially holds the interlock.

Type An integer describing the type of the interlock (currently not used).

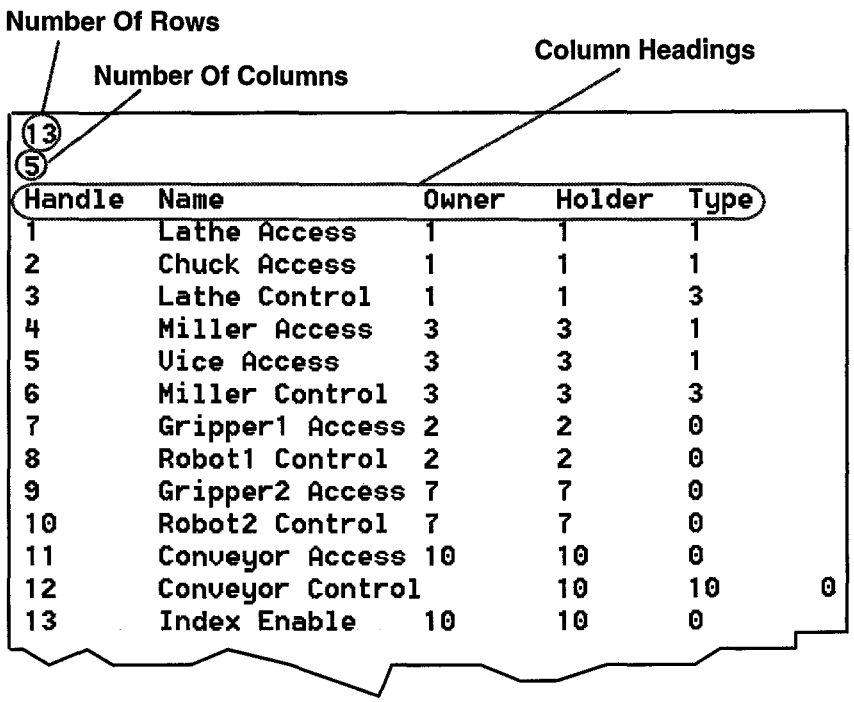


Figure 2 Structure of 'semaphrs.dat' File

Communications

The Denford CIM system uses the Client Server model in its inter-program communication system. Each device driver and cell manager is a DDEML server. For another program to link to either a driver or a cell manager to request its services it is necessary to use DDEML client type routines. The servers can be attached to using the following DDEML parameters. For the Service string use the Machine Name, the Topic string is 'data' and the Item string is 'status'. When connecting across a network using Net DDE the service string becomes '\\<Computer Network name>\NDDE$' and the topic string '$<Machine Name>'. For example, to connect to the service 'Lathe' which is executing on the computer called 'FMSPC' use a Service string of '\\FMSPC\NDDE$' and a topic string of 'Lathe$'.

Cell Controller - CASE Study

Introduction

The purpose of this CASE study is to demonstrate how to set up a DENFORD CIM system. In this case study a CIM system which consists of the following layout will be set up.

Installation

The Cell Control programs and Dynamically Linked Libraries are installed to a directory '\cim\bin'. The following DLLs must be present for the Cell Controller to function correctly.

DLL	Function
BC402RTL	Borland run time library
CONVSDLL	DDEML Client conversation class
CTL3D	Enables 3D look and feel to windows interfaces
DMTCOMMS	Generic communications class
DMTRS232	RS232 communication class
IMGRDLL	Interlock Database DLL
IOCARD	I/O Adaptor Card Interface
RES	Graphics resource

A table which is critical to the operation of the Denford CIM is the machine name table. This associates a unique identity number or 'handle' with each machine. Currently these values are hard wired.

Machine Number	Decimal Value	Machine Name
0	1	unused
1	2	Lathe
2	4	Robot1
3	8	Miller
4	16	CMM
5	32	AGV
6	64	Vision System
7	128	Robot2
8	256	Robot3
9	512	ASRS
10	1024	Conveyor

Machine Name Table

Another critical data file which must be present is the interlock database 'semaphrs.dat'. The structure of this file is described in the Cell Controller Manual under the section 'Interock Manager File Formats'. Here are the contents of this file:

```
8
5
Handle   Name      Owner
    Holder   Type
1   Lathe Access 1    1
    1
2   Chuck Access 1    1
    1
3   Lathe Control     1
    0    3
4   Miller Access     3
    3    1
5   Vice Access  3    3
    1
6   Miller Control    3
    0    3
7   Gripper1 Access   2
    2    0
8   Robot1 Control    2
    0    0
```

For a machine tool we create two extra interlocks after the control interlock, this one to access the machine.
The owner is the machine '1', the Lathe. This machine also holds the interlock when the system is first started. Type information is currently unused

The second interlock is to access the machines part clamp (chuck or vice)

Each machine **must** have a control interlock. Currently the control interlock is not used but will be required by future versions of the system

Two extra interlocks for the Miller

For a robot we create an extra interlock for accessing the gripper

Some of the interlocks in this file are referred to by their handle within the 'Ini' file of some device drivers. It is imperative that these references have the correct interlock handles.

Preparing Device Drivers and Machines

1 The Robot

Of all the device drivers the robot is the most complex to set up. To initialise the robot correctly two files have to be prepared, an 'Ini' file and a command file.

Here is a documented robot device driver 'Ini' file.

```
[IO PORT DATA]
```

```
Interface
Module
Port=Port4
```

This is an important entry. On the back of the Denford Interface Module there are several IO ports (see IO Port Interface - Denford Interface Module section of the Cell Controller Manual). A robot requires at least a four bit port. In this system there are no other machines that require a four or six bit port on the Interface Module so the user has a choice of settings in this option. Make sure however the robot cable is plugged into the correct port.

```
Idle Signal=0
```

This is a little-used entry whereby the user can specify what decimal value on the input port indicates that the robot is ready to execute another command. This defaults to zero. When the robot is ready to execute a command it sends an 'IDLE' signal back to the device driver by setting all the input port lines low (ie. a decimal value of zero). The robot program could be changed so that it sets all the input port lines high to indicate an idle state (ie. a decimal value of 15 on a 4 bit port or 63 on a 6 bit port). The advantage of this is that when the robot is not switched on the input port lines will all be low so the robot will appear to be 'BUSY' executing a command, after all if the robot is not powered up the device driver cannot request it to execute a command.

```
[SERVICE]
Machine
Name=Robot1
```

It is vital that this entry is set to one of the acceptable machine names listed in the Machine Name Table. In this case study there are several possible machine names for the robot ranging from 'Robot1' to 'Robot3'. As the robot is the first robot in the system the most logical name is 'Robot1'.

```
[TIMER]
Sampling
Interval = 1000
```

This figure dictates how often (in milliseconds) the robot device driver checks whether a change has occured on the input port. As there are only three machines being serviced by this cell PC this value can be set quite low.

```
[INTERLOCKS]
Gripper = 7
```

This figure **must** correspond to that specified in the interlock database (the 'semaphrs.dat' file).

```
Control = 8
```

This figure **must** correspond to that specified in the

```
6
Id  Signal  Description  -
-1  <-  Format Code  -
```

Robot command entries start on the next line onwards

```
1   1   Move to Hopper and
Take Billet  -
```

Command number 1
Activated by a signal value of 1 on the input port.
Description: "Move to Hopper and Take billet"

```
1   7   -   -
```

Take interlock 7 before executing the command. If the driver cannot take the interlocks (because another driver currently one) the command will fail.

```
1   7   -   -
```

Keep interlock 7 ('Gripper1 Access' see interlocks table) on terminating the command

```
2   2   Put part into Lathe  -
```

Command number 2
Activated by a signal value of 2 on the input port.
Description: "Move to Hopper and Take billet"

```
3   7   1   2
```

Take interlocks 7, 1 and 2 before executing the command.

```
3   7   1   0
```

Keep interlocks 7 and 1 on terminating the command.
Release interlock
Once a part has been placed in the chuck

```
3   3   Open gripper, leave
Lathe and Park   -
2   7   1   -
2   0   0   -
4   4   Into Lathe around Part
    -
2   7   1   -
2   7   1   -
5   5   Leave Lathe with part
and Park -
2   7   1   -   -
2   7   0   -   -
```

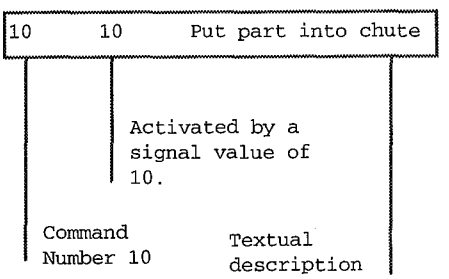
	interlock database (the 'semaphrs.dat' file).
[FILES]	
Status History = robot1.log	This file can have any name.
Commands File = latherob.cmd	This file can have any name.
[COMMAND TIMINGS]	This section contains command timings for simulations.
1=7	The command '1' takes 7 seconds to complete in simulation mode.
2=15	The command '2' takes 15 seconds to complete in simulation mode.
3=6	
4=5	
5=9	
6=4	

Note that it is irrelevant whether the robot has a linear slide or not because in the Denford CIM systems the slide is always slaved off the robot so the robot and slide are together treated as an independent unit.
An example program for this robot is shown in Appendix A. The robot program has several move numbers that are listed in the following table:

Move Number	**Binary Value**	**Robot action or command**
1	000001	Move to Hopper and Grab Billet
2	000010	Place Part in Lathe
3	000011	Release in lathe and park
4	000100	Move from park to Lathe
5	000101	Take Part from Lathe and Park
6	000110	Place part in Miller
7	000111	Leave Miller and park
8	001000	Go into Miller and grip Part
9	001001	Take part from Miller and park
10	001010	Place part in Chute

When the the robot program is executing the robot implements actions depending on signals sent to it via the IO Port Interface. The robot device driver sends the binary equivalent of the move number to the robot, via the IO Port Interface, to prompt it to perform an action. The robot device driver itself inputs its set of 'prompts' or robot commands from a command file.

```
6    6    Place  part  in  Miller
vice -
3    7    4    5    -
3    7    4    0    -
7    7    Open   gripper,   leave
Miller and Park  -
2    7    4    -
2    0    0    -
8    8    Into    Miller    around
Part -
2    7    4    -
2    7    4    -
9    9    Leave Miller with part
and Park -
3    7    4    5
3    7    0    0
10   10   Put part into chute   -
```




```
1    7    0    0
1    0    0    0
```

When setting the interlock requirements for commands be careful to release interlocks at the appropriate time otherwise two problems can occur
1 Deadlock - A machine takes an interlock but fails to release it, the next machine in the sequence which requires the same interlock is stuck because it cannot take it.
2 Safety Failure - Two machines could collide because an interlock was not inserted to prevent a dangerous situation occuring

```
[IO PORT DATA]
Interface Module Port =
PORT2A
```

All machine tools require a two bit port. In this system there are two machine tools a Lathe and a Miller. There are two 2 bit ports available on the Interface Module so the user has a choice of settings in this option. As a standard, the Lathe is always put in the first port and the Miller in the second port, this carries over to the RS232 serial ports as well. Therefore the Lathe is attached to Port 2A and Miller to 2B.

```
Clamp Bit=1
```

Here the bit which used to transmit and receive information about the part clamp (Chuck) is set.
Port 2A is a 2 bit port, the low bit is addressed bit '0' and the high bit as '1'.

```
Machining Cycle Bit=0
```

The Low bit is specified for use in transmitting and receiving information about the Machining Cycle. The output bit is used to prompt the machine tool to start a machining cycle. The input bit indicates whether the machine is in cycle or not

```
Machine Idle=1
```

Specifies the form of machine IDLE signal as being a 1 input on the Machining Cycle bit. It also indicates that to start the machine cycle a 1 must be output on this bit.

```
Clamp Open =1
```

Specifies that the part clamp is open when a 1 is input on the Clamp Bit. To open the Part Clamp a 1 must be output on this same bit.

```
[SERVICE]
Machine Name=Lathe
```

It is vital that this entry is set to one of the acceptable machine names listed in the Machine Name Table. In this case only 'Lathe' is available

```
[FILES]
Status History=lathe.log
```

Name of the event log file.

```
Commands File=lathe.cmd
```

Make sure the name of the command file is correct.

```
[FILE EXTENSIONS]
CNC Files=tap
```

This is required for manual program download. The only files listed for download purposes are those with this extension.

```
[COMMS]
Port=COM1
```

Specify that Lathe uses RS232 port 1

```
Settings        =4800,n,8,1
```

Baud rate is 4800, No parity, 8 data bits, 1 stop bit

```
Flags=H1
OutBufferSize=1024
InBufferSize=1024
TimeOut=        15000
```

The maximum time to wait in milliseconds for a NC block to be transmitted, in this case 15 seconds.

```
[TIMER]
Sampling Interval=600
[INTERLOCKS]
Part Clamp=2
```

Specify Part Clamp interlock handle (**this must correspond to the value specified for it in the Interlocks Database**)

```
Machine Access=1


[COMMAND TIMINGS]
1=1
2=1
3=7
4=1
5 ltest.tap=19
5 lking.tap=19
```

Specify machine Access interlock handle (**must be the correct value**)

Appendix A - Program for RV-M1 Robot Mounted On A Linear Slide

```
'*********************************************
' ROBOT WITH LINEAR SLIDE CONTROL PROGRAM
'
' Written by R. Bovill
' 18 January 1993
' (C) Denford Machine Tools
'
' Modifications...
' 16/2/93 - J.bond - GOTOs added to generate fms70 type commands.
' 3/3/93           - slide move when loading lathe added.
' 15/11/94 - R. Bovill - Made following changes:
' 396 GT 412 => 396 GT 20
' 590 GT 712 => 590 GT 20
' 690 GT 212 => 690 GT 20
' Robot always returns to park after a move
' 15/12/94 - R Bovill - Simplified command signal numbering convention
'                     - Speeded up robot and simplified move number convention
'*****************************************
' ROBOT POSITIONS
' PARK
' robot park          100
' LATHE
' square to            11
' approach part        12
' around part          13
' retract              15
' MILLER
' square to            21
' approach part        22
' almost around part   23
' around part          24
' retract              25
' HOPPER
' square to            31
' approach part        32
' around part          33
' retract              35
' CHUTE
' square to            41
' approach part        42
' around part          43
' retract              45
DL 1,2048
'
'
'1 - Move to Hopper and Grab Billet
'2 - Place Part in Lathe
'3 - Release in lathe and park
'4 - Move from park to Lathe
'5 - Take Part from Lathe and Park
'6 - Place part in Miller
'7 - Leave Miller and park
'8 - Go into Miller and grip Part
'9 - Take part from Miller and park
```

```
'10 - Place part in Chute

 10 NT
'Initialise the robot and slide
 15 GS 100
'The Main Loop
 20 OD 0
 25 ID
 30 EQ 1,600
 35 EQ 2,200
 40 EQ 4,300
 45 EQ 6,400
 50 EQ 8,500
 55 EQ 10,700
 65 GT 20

'****************************************
'Initialise the system...
'Move to park
100 MO 100, O
' Initialise the sliding base
105 GS 1202
' Set the gripper pressure
'110 GP 12,12,10
' Set the speed and acceleration
115 SP 9,H
150 RT

'****************************************
' COMMAND 2 - PUT PART INTO LATHE
' Debounce input
200 ID
204 NE 2,20
208 OD 2
' Park
212 MO 100, C
' Move slide to lathe
216 GS 1360
' Square up to machine
224 MO 11, C
'IF (loading lathe with linear slide)
' Move slide so robot moves towards chuck
'236 GS 1420
'ELSE
' Approach chuck
232 MO 12, C
236 SP 3,L
' Put part into chuck
240 MO 13, C
'ENDIF
' COMMAND 3 - LEAVE LATHE AND PARK
' Interlock until chuck is closed
' loop.
244 OD 0
248 ID
252 NE 3, 248
```

```
' Debounce input
256 TI 5
260 NE 3, 248
262 OD 3
' Open gripper
264 GO
266 TI 5
'IF (loading with slide)
' Move slide so robot moves away from chuck
' 272 GS 1360
'ELSE
' Move away from part
272 MO 15, 0
'ENDIF
' Out of machine
276 SP 9,H
' Back to facing the machine
```

Machine Tool Installation

The device drivers for both Denford lathes and millers communicate to the machine tools in two different ways: RS232 serial links and digital I/O signals.

RS232 is used for program download; Direct Numerical Control (DNC) and detailed state checking. An RS232 cable should connect from a serial port on the supervising Cell PC to the external serial port on the machine tool.

I/O signals are connected directly from the Interface Module to the machine's auxiliary input and output port. There are two input lines and two output lines which can be used to transmit low-level state information to the device driver.

RS232 serial communication

To communicate via the serial link, both machine tool and device driver must use the same settings. For the machine tool, the settings are stored in the 'options file' (.opt extension), the serial communications parameters have the prefix EXTCOMM.

EXTCOMM_ENABLE 1	enables serial communications
EXTCOMM_ROUTE 1	specifies the use of the com port
EXTCOMM_SDEVICE COM1	specifies the name of the com port
EXTCOMM_BAUD 4800	the baud rate (300, 1200, 2400, 4800, 9600)
EXTCOMM_PARITY 0	parity settings (0=none, 1=odd, 2=even)
EXTCOMM_DATABITS 8	specifies the number of data bits (7, 8)
EXTCOMM_STOPBITS 1	specifies the number of stop bits (1, 2)
EXTCOMM_LF 1	specifies whether Line Feed is used (0, 1)
EXTCOMM_ES 1	1st escape character
EXTCOMM_EE 2	2nd escape character

The address and interrupt request number for the machine's communications port is stored in the 'Go' file (.go extension).

Address, interrupt, name
$3E8 5 COM1

The name in the Go file must correspond to the name in the Options file.

Communications must use Hardware flow control (CTS/RTS).

For the device driver, the communications settings are stored in the Initialisation file (.ini extension).

[COMMS]			
Port	=	COM1	specifies which PC port to use
Settings	=	4800,n,8,1	the communication settings
Flags	=	H1	flow control flags
OutBufferSize	=	1024	buffer sizes (bytes)
InBufferSize	=	1024	
TimeOut	=	15000	RS232 timeout (ms)

The 'Port' name is Window's reference to the serial port to be used (see Window's Control Panel's Ports section). The 'Settings' specify the baud rate, parity (n=none, o=odd, e=even), data bits and stop bits. 'Flags' specify the flow control to be used, (H0=no handshaking, H1=hardware handshaking, H2=XON/XOFF flow control).

The settings for the machine and the device driver above do correspond so serial communications can take place.

There are two methods for transferring data to a Denford Machine controller, file transfer and Direct Numerical Control (DNC or drip-feeding).

File Transfer

The Denford controller recognises the command "$$MACHINE FROM <filename>", where 'filename' is a valid NC program. If this command is sent over the serial link the controller will load the specified program and immediately start execution. The file name holds a path to the program so, if the machine tool is part of a PC network, the path could specify a Network File-server.

DNC

The Denford controller also recognises the commands "$$MACHINE", "$$SCREEN" and "$$EXECUTE". These commands initiate drip-feeding of a NC program.

$$MACHINE executes each block as it is received.

$$SCREEN simulates each block as it is received.

$$EXECUTE executes each block in the current machine mode (AUTO or SIMULATE).

To drip feed a program, put the line '$$MACHINE' at the beginning of the file and the line '$$RESET' at the end. The '$$RESET' command flushes the RS232 buffer and resets the controller to its initial state.

I/O Signal Communication

In a Cell PC there is an Industrial I/O card. The I/O card provides 16 relay outputs and 16 photo-couple inputs. This allows the Cell PC to send signals directly to a machine tool via its auxiliary I/O port.

All Denford Machine Tools are connected to the I/O interface module in ports 2A or 2B; the lead has a QM style plug which fits into the auxiliary port. Some machines may already have an emergency stop terminator plug in the auxiliary port which allows the machine to operate without a remote emergency stop. When the machine is to be part of a CIM, the terminator plug should be replaced with the lead to the I/O interface module.

The auxiliary port provides two inputs to the machine tool, two outputs from the machine and an emergency stop circuit. The CIM software automatically energises the emergency stop circuit unless the remote emergency stop button is depressed. The inputs and outputs to and from the machine tool correspond to four 'M' codes:

M62 output 1 on
M63 output 2 on
M64 output 1 off
M65 output 2 off

There are four other 'M' codes which can be used to delay execution of a NC program until their condition is met:

M66 wait for input 1 to turn on
M67 wait for input 2 to turn on
M76 wait for input 1 to turn off
M77 wait for input 2 to turn off

Input/output 1 corresponds to bit 0 on the I/O interface module, input/output 2 corresponds to bit 1.

The initialisation file for a device driver contains a section headed '[IO PORT DATA]' which holds an interpretation of the IO signals. See the device driver manual for a detailed explanation.

Input 2 on a Denford Machine Tool can be used to remotely control the clamp. The clamp opens when a signal (+24V DC) is sent and closes

when it is removed. To allow the remote clamp to operate, the machine tool's options file (.opt extension) must have the line:

REMOTECLAMP 1 (MILLER)

REMOTECHUCK 1 (LATHE)

Structure of a CIM program for machine tools

To safely use a Denford Machine Tool in a CIM environment it is recommended that the programming technique explained below is used.

The NC program should be prefixed with the lines:

1. **M76** *(wait for input 1 to turn off)*
2. **M62** *(set output 1 to on)*
3. **M39** *(close the door)*

When this program is executed, it immediately halts at line 1 and waits for confirmation from an external signal (usually provided by the Denford device driver software). Line 2 sends a signal to the device driver to tell it that the machine is busy and line 3 closes the door ready for machining.

At the end of the program (line n) the 'busy signal' should be turned off with the lines:

n-2. **M38** *(open the door)*

n-1. **M64** *(set output 1 to off)*

n. **M30** *(end of program)*

The main program can either be directly inserted between the above two sections or it can be stored as a separately and called as a sub-program.

'M' code number 98 calls a subprogram, the parameter P holds the name of the program to be called. So in the example above, line 4 could be:

4. M98 P0003; *(call sub-program 0003)*

Program 0003 will perform the required actions and then execute the M

code 'M99' to return to the original calling program. Refer to the machine-tool programming manual for further help.

RS232 Functions of FANUC 0 Controllers

As well as the standard RS232 link capabilities of the FANUC 0 controls there are also a further two functions available for mass data transfer and specialist computer links.

DNC

By programming auxiliary code "M29" in "MDI" mode, The control will place itself in the "DNC" mode. Any valid NC blocks sent to the control will be executed but not stored in the controls memory and so a program of infinite length can be executed by the CNC machine.

The control will exit DNC mode if:

- An invalid block is transmitted.
- An "M02" or "M30" block is transmitted.
- Control/Reset is pressed.

During transmission the XON/XOFF protocol is used to control data flow.

"MINP" FUNCTION

With the "MINP" function enabled, the control will receive any valid NC block from a host computer and store these blocks in its internal memory. However, upon receipt of a program end code, it will automatically begin to execute the program just received until either:-

- "M02" or "M30" is executed, in which case the control will exit "MINP" mode.
- Auxiliary code "M15" is executed, in which case cycle execution will stop and RS232 input will resume.
- Control/Reset is pressed.

"MINP" mode us entered in one of two ways depending on the type of machine being used: note parameter PWE (Parameter Write Enable) should be set to 1 to alter the parameters.

1) **ON DENFORD FANUC 0 LATHES**

Set parameter 11, bit 7 to 1	use "MINP" input
Set diagnostic 452, bit 4 to 1	use "MINP" input
Set parameter 111 to 15	the M code M15 is not buffered

2) ON DENFORD FANUC 0 MILLERS

Set parameter 11, bit 7 to 1	use "MINP" input
Set diagnostic 451, bit 7 to 1	use "MINP" input
Set parameter 111 to 15	the M code M15 is not buffered

When parameter 11 is altered, PS alarm 00 occurs. This means that the controller should be switched off and on again with the new setting.

Please note that when the control starts to receive a NC program into its memory when in "MINP" mode, it will first erase any programs currently in its memory so the program protect key should be switched to OFF.

Baud rate is set with the BRATE1 parameter (552), note the I/O parameter must be set to 0 to use BRATE1.

BAUD : 4800
DATA : 7
STOP : 1
PARITY: EVEN

Again, data transmission is controlled using <XON> and <XOFF> protocol.

Start of program characters
"% ^M ^J" (CR/LF)

End of program characters
"M15 ^M ^J (CR/LF)
%"

All spaces are optional in a program, each line ends with a ^M and ^J (CR/LF)

Refer to the FANUC manual appendix for further details of parameters.

I/O Functions of FANUC 0 Controllers

The FANUC 0 controllers work in a similar way to the Denford controllers regarding auxiliary inputs and outputs, however various parameters must be set to enable them.

1) **ON DENFORD FANUC 0 LATHES**

Set parameter 452, bit 6 to 1 use auxiliary 2 for chuck
Set parameter 450, bit 7 to 0 enable M62/M64

2) **ON DENFORD FANUC 0 MILLERS**

Set parameter 450, bit 2 to 1 use auxiliary 2 for chuck
Set parameter 450, bit 7 to 0 enable M62/M64

RS232 Functions of FANUC 21i Controllers

As well as the standard RS232 link capabilities of the FANUC 21i controls there are also a further two functions available for mass data transfer and specialist computer links.

DNC

By programming auxiliary code "M29" in "MDI" mode, The control will place itself in the "DNC" mode. Any valid NC blocks sent to the control will be executed but not stored in the controls memory and so a program of infinite length can be executed by the CNC machine.

The control will exit DNC mode if:

- An invalid block is transmitted.
- An "M02" or "M30" block is transmitted.
- Control/Reset is pressed.

During transmission the XON/XOFF protocol is used to control data flow.

"MINP" FUNCTION

With the "MINP" function enabled, the control will receive any valid NC block from a host computer and store these blocks in its internal memory.

However, upon receipt of a program end code, it will automatically begin to execute the program just received until either:-

- "M02" or "M30" is executed, in which case the control will exit "MINP" mode.
- Auxiliary code "M15" is executed, in which case cycle execution will stop and RS232 input will resume.
- Control/Reset is pressed.

"MINP" mode us entered in one of two ways depending on the type of machine being used: note parameter PWE (Parameter Write Enable) should be set to 1 to alter the parameters.

1) **ON DENFORD FANUC 21i LATHES**

Set parameter 3201, bit 7 to 1	use "MINP" input
Set Keep Relay K1.4 to 1	use "MINP" input
Set parameter 3411 to 15	the M code M15 is not buffered

2) **ON DENFORD FANUC 21i MILLERS**

Set parameter 3201, bit 7 to 1	use "MINP" input
Set Keep Relay K1.4 to 1	use "MINP" input
Set parameter 3411 to 15	the M code M15 is not buffered

When parameter 3201 is altered, PS alarm 00 occurs. This means that the controller should be switched off and on again with the new setting.

Please note that when the control starts to receive a NC program into its memory when in "MINP" mode, it will first erase any programs currently in its memory so the program protect key should be switched to OFF.

To check the RS232 settings on the Fanuc 21i use the following procedure.

1. Press the **SYSTEM** Button.
2. Press the **SYSTEM SOFT** Button.
3. Press the Right hand extended menu Button three times so that the **ALL I/O** section is displayed.
4. Press the **SYSTEM SOFT** Button under the **ALL I/O** section and the RS232 setting will be displayed.

E.g.

I/O CHANNEL : 1
DEVICE NUM : 0
BAUD RATE : 9600
STOP BIT : 2

Again, data transmission is controlled using <XON> and <XOFF> protocol.

I/O CHANNEL : 1
DEVICE NUM : 0
BAUD RATE : 9600
STOP BIT : 2

Parameter 0103 = 10 (Baud rate 4800)
Parameter 0103 = 11 (Baud rate 9600)
Parameter 0103 = 12 (Baud rate 19200)
Parameter 0101.0 = 0 (1 stop bit)
Parameter 0101.0 = 1 (2 stop bits)

Again, data transmission is controlled using <XON> and <XOFF> protocol.

Start of program characters
"% ^M ^J" (CR/LF)

End of program characters
"M15 ^M ^J (CR/LF)
%"

All spaces are optional in a program, each line ends with a ^M and ^J (CR/LF)

Refer to the FANUC manual appendix for further details of parameters.

I/O Functions of FANUC 21i Controllers

The FANUC 21i controllers work in a similar way to the Denford controllers regarding auxiliary inputs and outputs, however various parameters must be set to enable them.

1) ON DENFORD FANUC 21i LATHES

Set Keep Relay 1.1 to 1 — use auxiliary 2 for chuck
Set Keep Relay 0.1 to 1 — Programmable chuck

2) ON DENFORD FANUC 21i MILLERS

Set Keep Relay 1.1 to 1 — use auxiliary 2 for vice
Set Keep Relay 0.0 to 1 — Programmable vice

3. **M62** *(set output 1 High / On)*

4. **M39** *(close the door)*

The door is closed in line 4 and line 3 sends a signal to the device driver to tell it that the machine is busy.

At the end of the program (line n-2) the 'busy signal' should be turned off with the lines:

n-3. **M38** *(open the door)*

n-2. **M64** *(set output 1 to off)*

n-1. **M15** *(resume "MINP" mode)*

n. *%* *(end of program)*

The main program can either be directly inserted between the above two sections or it can be stored as a separately and called as a sub-program.

'M' code number 98 calls a subprogram, the parameter P holds the name of the program to be called. So in the example above, line 4 could be:

4. M98 **P0003;** *(call sub-program 0003)*

Program 0003 will perform the required actions and then execute the M code 'M99' to return to the original calling program. Refer to the machine-tool programming manual for further help.

Robot Installation

Like machine tools, robots communicate with RS232 signals and digital I/O signals; however once a program has been downloaded to the robot, the RS232 cable can be removed because it is not used for sending or receiving status information while the robot is in cycle.

RS232 serial communication

The robot's control unit has an external RS232 port at the back; this should be connected to a computer which stores the robot program (either a cell computer or the host computer). The communications settings are altered with a set of switches in the side door on the right hand side. There are three sets of switches labelled SW1, SW2 and SW3, each is a bank of eight on/off switches where ON is signified by a switch in the upper position; here a 1 shall indicate ON (up) and a 0 shall indicate off (down).

Switch SW1 controls various operations but switch 1 controls the carriage return and line feed function for the RS232 operation. SW1 should be set to 01100001. Refer to the robot manual pages 2-15 and 2-16 for guidance.

Switch SW2 controls the asynchronous transmission including the number of stop bits, parity and character length. It should be set to: 01011110 which indicates 7 data bits, even parity and 1 stop bit. Refer to the robot manual Appendix 6 for guidance.

Switch SW3 controls the Baud rate and should be set to 00000010 which produces a baud rate of 4800.

To communicate to the robot, Windows Terminal should be used with the same settings, i.e. 4800 baud, 7 data bits, 1 stop bit, even parity and XON/XOFF flow control. The robot teach box should be switched OFF and the side setting switches (the two metallic toggle switches) should both be in the DOWN position to receive a program over the serial link.

The robot should now be able to respond to and receive commands. If 'WH' is typed into the Terminal window, the robot should reply with the positions of each of its axes (if it has been nested). To download a file,

choose **Transfers|Send Text File** from the menu of Terminal and select the robot file name to send. When the OK button is pressed the file will be sent to the robot. Either a program or a set of positions can be transferred using this procedure.

I/O Signal Communication

There are two commands in the robot language which send and receive information to and from the robot's external I/O port. The commands are: OD (Output Direct) and ID (Input Direct).

Output Direct sends data unconditionally through the output port. It can send a number between 0 and 255 (decimal) and will maintain that output until it is changed by another OD instruction.

Input Data unconditionally reads from the input port and places the result into the robot's internal comparison register. The data can then be used by the robot to compare with constants and to make program flow decisions.

Structure of a CIM program for robots (Using a Robot RS232 Device Driver)

Utilising the RS232 Robot Device Driver means that Robot programs are down loaded from the Cell controller, to the Robot controller, and are immediately executed as and when they are required during the CIM System cycle.

When a program is downloaded to the robot controller using an RS232 Robot Device Driver, the first thing that must be done is to delete the old program from the robot's memory. This can be done by prefixing the program with the line

DL 1, 2048

The basic Robot program structure should be as follows :-

DL 1,2048	Deletes any program left in the Robot controller
10 OB +0	Turns Output bit 0 ON (Busy signal)
20	Main body of program
1000 OB +0	Turns Output bit 0 OFF (Idle signal)
RN	Run the program immediately

Structure of a CIM program for robots (Using a Robot I/O Device Driver)

When a program is downloaded to the robot, the first thing that must be done is to delete the old program from the robot's memory. This can be done by prefixing the program with the line

DL 1, 2048

This deletes program lines 1 to 2048, i.e. all possible line numbers.

The program should be written as a set of subroutines, each performing a simple sequence of actions. The subroutines can be executed in response to an external I/O signal.

The Denford robot programs all have three distinct sections:

1. An initialisation sub-routine

2. A main program loop

3. A set of movement sequences

The initialisation routine is called once at the beginning of the robot program. It may nest the robot and set various movement parameters

such as speed, acceleration and gripper strength.

The main loop continually reads the input port for data and continually sets the output port to 0 (idle). It appears in the program for a robot servicing a milling machine like this:

```
20 OD 0
30 ID
40 EQ 1, 600
50 EQ 3,300
60 GT 20
```

line 20 sets the output port to 0 (idle)

line 30 reads the input port and stores the result in the robot's internal comparison register

line 40 compares the register to '1', if the comparison is true, then the program jumps to line 600

line 50 compares the register to '3', if the comparison is true, then the program jumps to line 300

line 60 finishes the loop and jumps back to line 20 to start again.

The two subroutines called do two separate tasks; the first loads a component into the machine, the second takes a component from the machine.

Each subroutine executes a simple sequence of movements to manipulate a component and move it to the correct place. A subroutine may also execute 'interlocks', for instance: the robot may take a component and place it in the jaws of a vice on a milling machine; the subroutine is now finished so it returns an idle signal to allow the vice to close. The robot is in a potentially dangerous position because the only safe command to execute would be to leave the milling machine and park. If another subroutine were to run, the robot would collide with the machine tool. To avoid the collision the subroutine does not return to the main program loop where any subroutine could be triggered, but it starts another loop which scans for an input which triggers the one safe subroutine to execute. An interlock like this can be seen at line 244 in the miller-robot program.

ASRS SETUP PROCEDURE
6.9.99
Newasrs1.doc (For ASRU18.EXE 20.5.99)

1. Stand Alone Setup

This ASRS set-up procedure is intended to be utilised when the ASRS is ready to be integrated in to a CIM system. (Note: - An axis override speed control unit has been fitted to the ASRS controller to help avoid any unwanted collisions between the end of arm tooling the ASRS framework.)

1.1 Powering Up

Connect the monitor and keyboard to the ASRS video and keyboard output ports. Plug the ASRS and monitor in to the mains supply. Ensure that the internal E-stop link is fitted between terminals 61 and 62 in the Control Cabinet see fig. 1.1.1. For location or that an I/O (E-stop) cable is fitted between the ASRS I/O Input and an active CIM Interface Module, or that an E-stop dummy plug has been fitted to the ASRS I/O socket. Insert the ASRS disk in to the floppy disk drive.

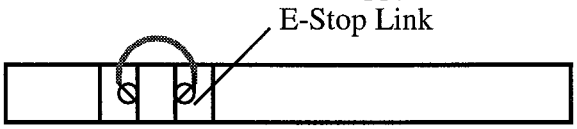


Fig 1.1.1

Depress the ASRS E-stop and switch on the ASRS. Wait for the ASRS software to activate.

1.2 Homing

When the ASRS is first powered up the software will automatically enter the Jog Mode. This is to ensure that the ASRS end of arm tooling can be "Jogged" to a safe position where it is able to travel to its 3 Home axis positions without the possibility of a collision with any part of the ASRS framework. Ensure that the ASRS E-stop is released and move the ASRS end of arm tooling to a safe position ready for homing, using the keys as detailed on the Jog Mode screen. Once in a safe position leave Jog Mode and select Home from the Main menu.
Home each of the 3 axes in turn starting with the X-axis, as detailed on the Home Mode screen.

1.3 Setting up Pallet location shelf Positions

After homing all the 3 ASRS axes. Position the Y-axis of the ASRS physically in the middle of the framework itself using JOG MODE (make a note of the Y axis value on the monitor). This means that the X, Y, Z dimension values are the same for both side 1 and side 2 of the ASRS when setting the ASRS POSITION for both sides (i.e. easier to set up). This position is set by entering the ASRS POSITION section of the SETTINGS option of the software. Once the ASRS POSITION is set for SIDE 1, ROW 1, COLUMN 1 (SRC) then by selecting the FIX PIGEONS option, this in effect calculates all side 1 ASRS POSITIONS if a COLUMN and ROW dimensions are entered in the ASRS.OPT file under (side 1 [step_1_x 260 and step_1_z 180] side 2 [step_2_x 260 and step_2_z 180]). Side 2 can be done just the same by selection

"Side 2 " at the appropriate time. The "T" option can be used here for inserting the displayed X, Y, Z dimension values as seen on the monitor. Example POSITION X=1404, Y=426, Z=955 for the side 1 top left pallet location shelve i.e. for SIDE 1, ROW 1, COLUMN 1.
The Z axis position for the ASRS POSITION needs to be taught with a pallet on the ASRS end of arm tooling so that the pallet top is in line with the top of the RSA where the pallet will eventually sit. This position is taught here because it is safe for any unscheduled movement in the Y-axis and also that the ASRS "PICK" routine is simple to construct with the first movement being on the Y-axis only.

1.4 Pick/Place Pallet location shelf Sequences

The "Pick" and "Place" sequences refer to the loading and unloading of a pallet on to and off the ASRS end of arm tooling relative to the pallet location shelf positions.
To input a "Pick" sequence select Setup, ASRS, Pallet location shelf, Load from the menu, initially this will show a blank box. Press Insert to produce the first line to edit and press Enter to edit it. Another box will be produced on screen with the three main axes X, Y and Z the axis W will be shown if the end of arm tooling wrist is articulated (optional) and D, which is the feed rate when positive and a dwell time in seconds when negative. Move to the required field in this list using the up and down arrow keys and input the sequence move, press Esc. To enter the line in to the sequence list, repeat this process to build up a "Pick" sequence. Note that the maximum number of lines allowed in a sequence is twelve. Refer to Appendix II for an example of a load sequence.
The "Place" sequence is entered in the same way as the "Pick" sequence from the Select, Pallet location shelf, Unload menu. Refer to Appendix II for an example of an "Place" sequence. Note that a separate "Pick" and "Place" sequence will have to entered for each side if the ASRS is double sided. See appendix 111 for example and additional information

1.5 Conveyor Position Setup

Setup the Conveyor position by entering Jog Mode and moving the end of arm tooling to a position inside the conveyor cut out loading area. Ensure that the top of the end of arm tooling is at least 5mm below the top face of the conveyor belting (i.e. X=0, Y=0 and Z=0). Note the position of the X Y Z coordinates or use the "T" on the keyboard and enter them at the Setup, Conv, Position option in the Main menu.

1.6 Conveyor "Place" Sequence Setup

The Conveyor "Place" sequence refers to the unloading of a pallet on to the conveyor from the end of arm tooling. It requires that the ASRS axes move from a position around 200mm vertically above the **Conveyor Position**. It then places the end of arm tooling in to the cut out section of the conveyor, waits for a few seconds for the pallet to be taken away by the conveyor belting and returns to the original position of around 200mm vertically above the conveyor. All positions are relative to the Conveyor "Place" position (i.e. X=0, Y=0 and Z=0). The sequence is entered and saved in the same way as the Pallet location shelf sequences, for an example of a Conveyor "Place" sequence see below
E.g. of a Conveyor "PLACE" routine

X0	Y0	Z200	D2000	200mm vertically above CONVEYOR POSITION
X0	Y0	Z 0	D2000	CONVEYOR POSITION (5 mm below conveyor belt surface)

1.7 Conveyor "Pick" Sequence Setup

The Conveyor "Pick" sequence refers to the ASRS moving the end of arm tooling from a position of around 200mm vertically above the Conveyor position into the cut out section of the conveyor where the **Conveyor Position** was originally set. All moves in the sequence list are relative to the **Conveyor Position** (i.e. X=0, Y=0 and Z=0). The sequence is entered and saved in the same way as Conveyor "Place" sequence. For an example of a Conveyor "Pick" sequence see below.

e.g. of a Conveyor "PICK" routine

X0	Y0	Z0	D-5	Dwell for 5 seconds to allow pallet to clear end of arm tooling
X0	Y0	Z200	D2000	200mm vertically above CONVEYOR POSITION

1.8 AGV Position Setup

The **AGV Position** needs to be taught with the top face of the pallet end of arm tooling sited around 15mm below where the pallet is actually going to sit on the AGV pallet shelving. (Ideally the Y-axis position should be the same as that set up when setting **ASRS POSITION** above).

1.9 AGV "Place" Sequence Setup

The AGV "Place" sequence refers to the unloading of a pallet on to the AGV pallet shelf from the ASRS end of arm tooling. It requires that the ASRS axes move from a position around 200mm vertically above the **AGV Position**. The end of arm tooling then drops vertically down to around 20mm below the AGV pallet shelving and thus deposits the pallet on the AGV shelving. The end of arm tooling is then withdrawn from under the pallet, along the Y-Axis of the ASRS for about 250mm to give adequate clearance for a 200mm pallet. The end of arm tooling is then brought vertically upwards and then back to its original starting position of 200mm above the **AGV Position**. All positions are relative to the AGV "Place" position (i.e. X=0, Y=0 and Z=0). The sequence is entered and saved in the same way as the Pallet location shelf sequences, for an example of an AGV "Place" sequence see below
e.g. of a AGV "PLACE" routine

e.g. of a AGV PLACE routines

X0	Y0	Z200	D2000	200 mm above AGV POSITION
X0	Y0	Z 0	D2000	AGV POSITION (15 mm below frame)
X0	Y-230	Z0	D2000	Back on Y axis to clear 200 mm pallet

				size)
X0	Y-230	Z200	D2000	200 mm above AGV POSITION offset 230 mm in Y axis
X0	Y0	Z200	D2000	200 mm above AGV POSITION

1.10 AGV "Pick" Sequence Setup

The AGV "Pick" sequence refers to the ASRS moving the end of arm tooling from a position of around 200mm vertically above the AGV position into the cut out section of the AGV where the **AGV Position** was originally set. All moves in the sequence list are relative to the **AGV Position** (i.e. X=0, Y=0 and Z=0). The sequence is entered and saved in the same way as AGV "Place" sequence. For an example of an AGV "Pick" sequence see below.

e.g. of a AGV PICK routines

X0	Y0	Z200	D2000	200 mm above AGV POSITION
X0	Y-230	Z 200	D2000	200 mm above AGV POSITION offset 230 mm in Y axis
X0	Y-230	Z0	D2000	Back on Y axis to clear 200 mm pallet size)
X0	Y0	Z0	D2000	AGV POSITION (15 mm below frame)
X0	Y0	Z200	D2000	200 mm above AGV POSITION

2 Local Command Control

2.1 Moving to a Safe Position

Before a local command can be carried out the end of arm tooling must be jogged to a safe position, this is to prevent a collision. A safe position is anywhere within a bounding box defined in the ASRS.opt file. Any axis in an unsafe position will have is decimal position value high lighted with a RED banner, this RED banner will disappear when the offending axis is Jogged to a safe position

2.2 Testing Pallet location shelf and Station Positions

In order to test the Pallet location shelf and Station positions the Move command is used. This command simply moves the ASRS end of arm tooling from its present position to the position defined in the command line. Caution should be taken when using this command as the end of arm tooling will take the shortest route between the two points without regard for the ASRS frame and a collision may occur.

A number defines each Pallet location shelf and Station as follows :-

ASRS (Pallet location shelves)	0
Side	1 or 2 (i.e. 1 or 2 sided ASRS)
Row	Range 1 to 5
Column	Range 1 to 7

E.g. PICK ASRS 1 1 1 - Move to ASRS, side 1, row 1, column 1

Conveyor	1
AGV	2

The PICK and PLACE commands can be carried out from the Local Option in the Main Menu. Select this option and type in the command as described above, to move to the desired position. The positions can then be tested to ensure that the end of arm tooling is moving to the correct co-ordinates, as defined in the Setup procedures. (Note: Use the axis override speed control override knob to help avoid any collision with the ASRS framework.)

3 Remote Command Control

3.1 Connection to CIM Cell

Before an ASRS can operate remotely it must have its E-stop/ Control and RS232 cables connected to the CIM Cell (i.e. Interface Module and PC serial port). In order for the E-Stop line to be effective the external E-Stop link bypass should be removed, see Fig 1.1 for location of link. The E-Stop/Control cable fits between the I/O socket connector on the ASRS and usually to a two-bit port on the Cell Interface Module (i.e. port 2A or 2B). The RS232 cable is connected between the ASRS RS232 connector and a serial port on the Cell PC (i.e. COM1, COM2 etc.)

3.2 Communication Settings

To enable the E-Stop/Control line to work, the Cell Interface Module port must be set to the correct port in the ASRS.ini file, in the CIM\OTHER or CIM\CELL1 directory, on the Cell PC. The RS232 link must also be set in this file, defining the RS232 port, Baud Rate, Data Bits, Stop Bits, Parity, Flags and In and Out Buffer size, see example Fig 3.2. The Flag setting defines the flow control of the RS232 link i.e. H2 is software control - Xon/Xoff, H1 is hardware control and H0 is no flow control. The Setting command defines the Baud Rate, Parity, Data Bits and Stop Bits, in that order. Parity is defined as e = Even, o = Odd and n = No.

```
[IO PORT DATA]                -       -
Interface Module Port = Port2A
[SERVICE]                     -       -
Machine Name                  =       ASRS
[FILES]                       -       -
Status History                =       asrs.log
Commands File                 =       asrs.cmd
[COMMS]
Port                          =       COM1
Settings                      =       4800,e,7,2
Flags                         =       H2
OutBufferSize                 =       128
InBufferSize                  =       128
```

Fig 3.2.1 ASRS.ini File (part) - RS232 Settings

The RS232 setting defined in the ASRS.ini file must be defined similarly in the ASRS.opt file on the ASRS floppy disk. Again these setting define Baud rate, Parity,

etc. and must match those settings defined in the ASRS.ini file for correct RS232 communication to be established, see example Fig. 3.2.2.

```
EXTCOMM_ENABLE 1
EXTCOMM_ROUTE 1
EXTCOMM_SDEVICE MASTER
EXTCOMM_BAUD 4800
EXTCOMM_PARITY 2
EXTCOMM_DATABITS 7
EXTCOMM_STOPBITS 1
EXTCOMM_LF 1
EXTCOMM_ES 1
EXTCOMM_EE 2
```

Fig. 3.2.2 ASRS.opt File (part) - RS232 Settings

To test that RS232 communications is working correctly, place ASRS in REMOTE mode, selected from the main menu, the ASRS end of arm tooling will have to be in a safe position first, refer to section 2.1 Moving to a Safe Position. Select the ASRS device driver on the Cell PC and ensure that the ASRS Status is idle, if not check that the E-Stop/Control cable connections and the ASRS.ini file is correct. If the Status on the device driver still reads Busy or Local Control only after this, check that the Control line pull-up resisters have been fitted in the ASRS.
Once the ASRS device driver Status is Idle, type in the device drivers command line, a simple command to ensure there is RS232 communication. Type for example PICK ASRS 1 1 1, which will move the end of arm tooling to ASRS side 1, column 1, row 1, without carrying out any "Place" sequences, this assumes that this position has been set for this particular pallet location shelf. If the end of arm tooling does not move, check that the ASRS has received the command, which will be displayed in the bottom left hand corner of the screen. Again if this fails to work correctly, ensure that RS232 settings in both the ASRS.opt and ASRS.ini files match. This failing check that the Address and Interrupt Request levels (IRQ) have been set correctly, for the ASRS in the ASRS.go file and for the Cell PC in Settings, Control Panel, System, Device Manager, Ports (COM & LPT) in Windows 95.
NOTE: There have been some problems with the RS232 communications in relation to Apricot Computers (Cell PC). The IRQ of the serial port is not recognised unless it is set again to the level it is displaying, i.e. even though it may be displaying the correct IRQ level it does not recognise it and has to be set again.

3.3 Device Driver Control

To ensure correct control of the ASRS from the Device Driver the ASRS.ini file should be set up to reflect the dimensions of the ASRS, in relation to the number of sides, rows and columns. The name of the inventory file should also be included and a list of the ASRS stations, see Fig. 3.3.1 for example.

```
[ASRS]
Rows=5
Columns=5
Sides=1
InventoryFile = invent.dat
ASRS=0
Conveyor=1
AGV=2
```

Fig. 3.3.1 ASRS.ini (part) - Dimensions and Station Settings

With the latest version of the ASRS device driver (April 1996 onwards) the command line no longer accepts ASRS Local type commands in the form, From *station* to *station*, where the station is a pallet location shelf. Instead it requires a Part name, the location of that part is looked up in the invent.dat file and the relavent position sent with the rest of the command to ASRS. For this reason the invent.dat file should be written in a grid format to represent what is held in each pallet location shelf, see Fig. 3.3.2 for an example on an invent.dat.

part1	part2	part3	part1	part2
part1	part2	part3	part1	part2
part1	part2	part3	part1	part2
part1	part2	part3	part1	part2
part1	part2	part3	part1	part2

Fig. 3.3.2 Invent.dat

Appendix I

The main reason for wanting to edit the ASRS.opt file to change the pallet location shelf positions, is speed. Using the ASRS software to enter all the pallet location shelf positions is time consuming and it is easy to loose track of the position to be entered. Below is an example of a 1 sided, 4 column by 5 row ASRS.opt file (part), where each pallet location shelves X,Y,Z co-ordinates are reprsented in the form *side_column_row_axis co-ordinates.* In order to edit these co-ordinates use an editor or word processor that has Find and Replace options in the Edit menu, such as Microsoft WordPad (i.e. Notepad does not have this facility). As each column has the same Z co-ordinates the existing Z co-ordinates can be found and then replaced by selecting the Replace option in the Edit menu, typing in the value that is already there in the Find box, then typing in the replacement in the Replace box. Taking the example below, all the Z co-ordinates for column 1 equal 930.00, so by finding the Z co-ordinates for each of the four columns all the Z co-ordinates can be changed. This procedure can be repeated for rows and X co-ordinates. Note that all Y co-ordinates are the same and therefore can be replaced in a single Find and Replace operation.

Appendix II

```
;
; options file for the
ASRS (ASRU18.EXE
20.5.99)
;
-------------------------
; Disable the test menu
option
TEST 0
; Enable the feed
override pot
NOPOTS 0
; Disable graphics
screen mode
SHOWINTEXT 1
-------------------------
; External RS233
communication
parameters
EXTCOMM_ENABL
E 1
EXTCOMM_ROUTE
1
EXTCOMM_SDEVIC
E MASTER
EXTCOMM_BAUD
4800
EXTCOMM_PARITY
2
EXTCOMM_DATAB
ITS 7
EXTCOMM_STOPBI
TS 1
EXTCOMM_LF 1
EXTCOMM_ES 1
EXTCOMM_EE 2
-------------------------
; CANDATUM 1
IOBASE $300
STEPSPERMM 274.1
ACCEL 1600
RAMP 0
HIGHFEED 30000
DATUMFEED 5000
MOVEFEED 20000
JOGFEED 5000
LIMIT_X 3050
LIMIT_Y 800
LIMIT_Z 1000
; MAXDATUM_X 0
MAXDATUM_Y 1
MAXDATUM_Z 1
DATUMKEY_Z 0
GAIN_X 0.2
GAIN_Y 0.2
GAIN_Z 0.2
rem KV_X 1
rem KV_Y 1
rem KV_Z 1
rem KR_X 3
rem KR_Y 3
rem KR_Z 3
KF_X 1
KF_Y 1
KF_Z 1
rem KI_X 0
rem KI_Y 0
rem KI_Z 0
step_1_x 260
step_1_z -180
step_2_x 260
step_2_z -180
-------------------------
; Safe Zone
parameters...
SAFELOW_X 0
SAFEHIGH_X 3050
SAFELOW_Y 395
SAFEHIGH_Y 460
SAFELOW_Z 0
SAFEHIGH_Z 1000
-------------------------
; No of (axes) :- 3 =
gantry, 4 = gantry +
wrist
AXES 3
-------------------------
; ASRS Physical make
up
ROWS 5
COLUMNS 5
SIDES 2
-------------------------
;Number of Stations
S 2
;Names of stations
S_1 Conveyor
S_2 Agv
;S_3
Unload_Conveyor
============
; ASRS
============

;Side 1 Pigeon Holes
1_1_1_X 1402.00
1_1_1_Y 426.00
1_1_1_Z 955.00
1_1_2_X 1662.00
1_1_2_Y 426.00
1_1_2_Z 955.00
1_1_3_X 1922.00
1_1_3_Y 426.00
1_1_3_Z 955.00
1_1_4_X 2182.00
1_1_4_Y 426.00
1_1_4_Z 955.00
1_1_5_X 2442.00
1_1_5_Y 426.00
1_1_5_Z 955.00
1_2_1_X 1402.00
1_2_1_Y 426.00
1_2_1_Z 775.00
1_2_2_X 1662.00
1_2_2_Y 426.00
1_2_2_Z 775.00
1_2_3_X 1922.00
1_2_3_Y 426.00
1_2_3_Z 775.00
1_2_4_X 2182.00
1_2_4_Y 426.00
1_2_4_Z 775.00
1_2_5_X 2442.00
1_2_5_Y 426.00
1_2_5_Z 775.00
1_3_1_X 1402.00
1_3_1_Y 426.00
1_3_1_Z 595.00
1_3_2_X 1662.00
1_3_2_Y 426.00
1_3_2_Z 595.00
1_3_3_X 1922.00
1_3_3_Y 426.00
1_3_3_Z 595.00
1_3_4_X 2182.00
1_3_4_Y 426.00
1_3_4_Z 595.00
1_3_5_X 2442.00
1_3_5_Y 426.00
1_3_5_Z 595.00
1_4_1_X 1402.00
1_4_1_Y 426.00
1_4_1_Z 415.00
1_4_2_X 1662.00
1_4_2_Y 426.00
1_4_2_Z 415.00
1_4_3_X 1922.00
1_4_3_Y 426.00
1_4_3_Z 415.00
1_4_4_X 2182.00
1_4_4_Y 426.00
1_4_4_Z 415.00
1_4_5_X 2442.00
1_4_5_Y 426.00
1_4_5_Z 415.00
1_5_1_X 1402.00
1_5_1_Y 426.00
1_5_1_Z 235.00
1_5_2_X 1662.00
1_5_2_Y 426.00
1_5_2_Z 235.00
1_5_3_X 1922.00
1_5_3_Y 426.00
1_5_3_Z 235.00
1_5_4_X 2182.00
1_5_4_Y 426.00
1_5_4_Z 235.00
1_5_5_X 2442.00
1_5_5_Y 426.00
1_5_5_Z 235.00
;Side 2 Pigeon Holes
2_1_1_X 1402.00
2_1_1_Y 426.00
2_1_1_Z 955.00
2_1_2_X 1662.00
2_1_2_Y 426.00
2_1_2_Z 955.00
2_1_3_X 1922.00
2_1_3_Y 426.00
2_1_3_Z 955.00
2_1_4_X 2182.00
2_1_4_Y 426.00
2_1_4_Z 955.00
2_1_5_X 2442.00
2_1_5_Y 426.00
2_1_5_Z 955.00
2_2_1_X 1402.00
2_2_1_Y 426.00
2_2_1_Z 775.00
2_2_2_X 1662.00
2_2_2_Y 426.00
2_2_2_Z 775.00
2_2_3_X 1922.00
2_2_3_Y 426.00
2_2_3_Z 775.00
2_2_4_X 2182.00
2_2_4_Y 426.00
2_2_4_Z 775.00
2_2_5_X 2442.00
2_2_5_Y 426.00
2_2_5_Z 775.00
2_3_1_X 1402.00
2_3_1_Y 426.00
2_3_1_Z 595.00
2_3_2_X 1662.00
2_3_2_Y 426.00
2_3_2_Z 595.00
2_3_3_X 1922.00
2_3_3_Y 426.00
2_3_3_Z 595.00
2_3_4_X 2182.00
2_3_4_Y 426.00
2_3_4_Z 595.00
2_3_5_X 2442.00
2_3_5_Y 426.00
2_3_5_Z 595.00
2_4_1_X 1402.00
2_4_1_Y 426.00
2_4_1_Z 415.00
2_4_2_X 1662.00
2_4_2_Y 426.00
2_4_2_Z 415.00
2_4_3_X 1922.00
2_4_3_Y 426.00
2_4_3_Z 415.00
2_4_4_X 2182.00
2_4_4_Y 426.00
2_4_4_Z 415.00
2_4_5_X 2442.00
2_4_5_Y 426.00
2_4_5_Z 415.00
2_5_1_X 1402.00
2_5_1_Y 426.00
2_5_1_Z 235.00
2_5_2_X 1662.00
2_5_2_Y 426.00
2_5_2_Z 235.00
2_5_3_X 1922.00
2_5_3_Y 426.00
2_5_3_Z 235.00
2_5_4_X 2182.00
2_5_4_Y 426.00
```

2_5_4_Z 235.00
2_5_5_X 2442.00
2_5_5_Y 426.00
2_5_5_Z 235.00
;Number of steps to load and unload side 1
A_1_OFF 4
A_1_ON 5
;Number of steps to load and unload side 2
A_2_OFF 5
A_2_ON 4
;ASRS side 1 Load & Unload sequences
A_1_ON_1_X 0.00
A_1_ON_1_Y 0.00
A_1_ON_1_Z 0.00
A_1_ON_1_D 10000.0
A_1_ON_1_W 0
A_1_ON_2_X 0.00
A_1_ON_2_Y 0.00
A_1_ON_2_Z -19.00
A_1_ON_2_D 3000.0
A_1_ON_2_W 0
A_1_ON_3_X 0.00
A_1_ON_3_Y -240.00
A_1_ON_3_Z -19.00
A_1_ON_3_D 3000.0
A_1_ON_3_W 0
A_1_ON_4_X 0.00
A_1_ON_4_Y -240.00
A_1_ON_4_Z 5.00
A_1_ON_4_D 3000.0
A_1_ON_4_W 0
S_1_X 506.00
S_1_Y 426.00
S_1_Z 78.00
S_1_OFF 2
S_1_ON 2
S_2_X 3031.00
S_2_Y 426.00
S_2_Z 68.00
S_2_OFF 5
S_2_ON 5
S_3_X 0.00
S_3_Y 0.00
S_3_Z 0.00
S_3_OFF 0
S_3_ON 0
S_4_X 0.00
S_4_Y 0.00
S_4_Z 0.00
S_4_OFF 0
S_4_ON 0
A_1_OFF_1_X 0.00
A_1_OFF_1_Y 0.00
A_1_OFF_1_Z 0.00
A_1_OFF_1_D 0.0
A_1_OFF_1_W 0
A_1_OFF_2_X 0.00
A_1_OFF_2_Y 0.00
A_1_OFF_2_Z 0.00
A_1_OFF_2_D 0.0
A_1_OFF_2_W 0
A_1_OFF_3_X 0.00
A_1_OFF_3_Y 0.00
A_1_OFF_3_Z 0.00
A_1_OFF_3_D 0.0
A_1_OFF_3_W 0
A_1_OFF_4_X 0.00
A_1_OFF_4_Y 0.00
A_1_OFF_4_Z 0.00
A_1_OFF_4_D 0.0
A_1_OFF_4_W 0
A_2_OFF_1_X 0.00
A_2_OFF_1_Y 0.00
A_2_OFF_1_Z 0.04
A_2_OFF_1_D 10000.0
A_2_OFF_1_W 0
A_2_OFF_2_X 0.00
A_2_OFF_2_Y 240.00
A_2_OFF_2_Z 0.00
A_2_OFF_2_D 3000.0
A_2_OFF_2_W 0
A_2_OFF_3_X 0.04
A_2_OFF_3_Y 240.00
A_2_OFF_3_Z -19.00
A_2_OFF_3_D 3000.0
A_2_OFF_3_W 0
A_2_OFF_4_X 0.04
A_2_OFF_4_Y 0.00
A_2_OFF_4_Z -19.00
A_2_OFF_4_D 3000.0
A_2_OFF_4_W 0
A_2_ON_1_X 0.00
A_2_ON_1_Y 0.00
A_2_ON_1_Z 0.00
A_2_ON_1_D 0.0
A_2_ON_1_W 0
A_2_ON_2_X 0.00
A_2_ON_2_Y 0.00
A_2_ON_2_Z 0.00
A_2_ON_2_D 0.0
A_2_ON_2_W 0
A_2_ON_3_X 0.00
A_2_ON_3_Y 0.00
A_2_ON_3_Z 0.00
A_2_ON_3_D 0.0
A_2_ON_3_W 0
A_2_ON_4_X 0.00
A_2_ON_4_Y 0.00
A_2_ON_4_Z 0.00
A_2_ON_4_D 0.0
A_2_ON_4_W 0
S_1_OFF_1_X 0.00
S_1_OFF_1_Y 0.00
S_1_OFF_1_Z 200.00
S_1_OFF_1_D 10000.0
S_1_OFF_1_W 0
S_1_OFF_2_X 0.00
S_1_OFF_2_Y 0.00
S_1_OFF_2_Z 0.00
S_1_OFF_2_D 3000.0
S_1_OFF_2_W 0
S_1_ON_1_X 0.00
S_1_ON_1_Y 0.00
S_1_ON_1_Z 0.00
S_1_ON_1_D -2.0
S_1_ON_1_W 0
S_2_OFF_1_X 0.00
S_2_OFF_1_Y 0.00
S_2_OFF_1_Z 200.00
S_2_OFF_1_D 10000.0
S_2_OFF_1_W 0
S_2_OFF_2_X 0.00
S_2_OFF_2_Y 0.00
S_2_OFF_2_Z -10.00
S_2_OFF_2_D 3000.0
S_2_OFF_2_W 0
S_2_OFF_3_X 0.00
S_2_OFF_3_Y -220.00
S_2_OFF_3_Z -10.00
S_2_OFF_3_D 3000.0
S_2_OFF_3_W 0
S_2_OFF_4_X 0.00
S_2_OFF_4_Y -220.00
S_2_OFF_4_Z 200.00
S_2_OFF_4_D 3000.0
S_2_OFF_4_W 0
S_2_OFF_5_X 0.00
S_2_OFF_5_Y 0.00
S_2_OFF_5_Z 200.00
S_2_OFF_5_D 3000.0
S_2_OFF_5_W 0
S_2_ON_1_X 0.00
S_2_ON_1_Y 0.00
S_2_ON_1_Z 200.00
S_2_ON_1_D 10000.0
S_2_ON_1_W 0
S_2_ON_2_X 0.00
S_2_ON_2_Y -220.00
S_2_ON_2_Z 200.00
S_2_ON_2_D 3000.0
S_2_ON_2_W 0
A_1_ON_5_X 0.00
A_1_ON_5_Y 0.00
A_1_ON_5_Z 0.00
A_1_ON_5_D 3000.0
A_1_ON_5_W 0
A_2_OFF_5_X 0.00
A_2_OFF_5_Y 0.00
A_2_OFF_5_Z 0.00
A_2_OFF_5_D 3000.0
A_2_OFF_5_W 0
S_2_ON_3_X 0.00
S_2_ON_3_Y -220.00
S_2_ON_3_Z -10.00
S_2_ON_3_D 3000.0
S_2_ON_3_W 0
S_2_ON_4_X 0.00
S_2_ON_4_Y 0.00
S_2_ON_4_Z -10.00
S_2_ON_4_D 3000.0
S_2_ON_4_W 0
S_2_ON_5_X 0.00
S_2_ON_5_Y 0.00
S_2_ON_5_Z 200.00
S_2_ON_5_D 3000.0
S_2_ON_5_W 0
S_1_ON_2_X 0.00
S_1_ON_2_Y 0.00
S_1_ON_2_Z 200.00
S_1_ON_2_D 3000.0
S_1_ON_2_W 0

Appendix III

Examples of a Conveyor "Pick" & "Place" sequences
E.g. of a Conveyor "PLACE" routine

X0	Y0	Z200	D2000	200mm vertically above CONVEYOR POSITION
X0	Y0	Z 0	D2000	CONVEYOR POSITION (5 mm below conveyor belt surface)

G.g. of a Conveyor "PICK" routine

X0	Y0	Z0	D-5	Dwell for 5 seconds to allow pallet to clear end of arm tooling
X0	Y0	Z200	D2000	200mm vertically above CONVEYOR POSITION

The D -3 in the above example denotes a dwell time in seconds; positive D values represent feed rates.

Appendix IV
General Notes

After homing all the 3 ASRS axes. Position the Y-axis of the ASRS physically in the middle of the framework itself using JOG MODE (make a note of the Y axis value on the monitor). This means that the X, Y, Z dimension values are the same for both side 1 and side 2 of the ASRS when setting the ASRS POSITION for both sides (i.e. easier to set up). This position is set by entering the ASRS, POSITION section of the SETTINGS option of the software. Once say the ASRS POSITION is set for SIDE 1, ROW 1, COLUMN 1 (SRC) then by selecting the FIX PIGEONS option, this in effect calculates all side 1 ASRS POSITIONS if a COLUMN and ROW dimensions are entered in the ASRS.OPT file under (side 1 [step_1_x 260 and setp_1_z 180] side 2 [step_2_x 260 and setp_2_z 180]). Side 2 can be done just the same by selection "Side 2 " at the appropriate time. The "T" option can be used here for inserting the displayed X, Y, Z dimension values as seen on the monitor. Example POSITION X=1404, Y=426, Z=955 for the side 1 top left pigeon hole i.e. for SIDE 1, ROW 1, COLUMN 1.

The Z axis position for the ASRS POSITION needs to be taught with a pallet on the ASRS manipulator arm so that the pallet top is in line with the top of the RSA where the pallet will eventually sit. This position is taught here because it is safe for any unscheduled movement in the Y-axis and also the ASRS PICK routine is simple to construct with the first movement being on the Y-axis only.

The CONVEYOR POSITION needs to be taught with the pallet manipulator arm actually sited in the conveyor as if loading a part onto it. (ideally the Y axis position should be the same as that set up when setting ASRS POSITION in 1. above).

The AGV POSITION needs to be taught with the top face of the pallet manipulator arm sited around 15mm below where the pallet is actually going to sit on the AGV framework. (ideally the Y axis position should be the same as that set up when setting ASRS POSITION in 1. above).

The CONVEYOR PICK & PLACE routines are set RELATIVE to the CONVEYOR POSITION which is X=0, Y=0 and Z=0, therefore a safe position vertically above the CONVEYOR POSITION would be X=0, Y=0, Z=200. Note the D value in the routine setting section is a speed value, 2000 is an average speed.

The AGV PICK & PLACE routines are set RELATIVE to the AGV POSITION which is X=0, Y=0 and Z=0, therefore a safe position vertically above the AGV POSITION would be X=0, Y=0, Z=200. Note the D value in the routine setting section is a speed value, 2000 is an average speed. In the case of the AGV the safe position of X=0, Y=0, Z=200 must be reached by a series of moves which steer clear of the loaded pallet using the Y and Z axes.

e.g. of a AGV PICK routines

X0	Y0	Z200	D2000	200 mm above AGV POSITION
X0	Y-230	Z 200	D2000	200 mm above AGV POSITION offset 230 mm in Y axis
X0	Y-230	Z0	D2000	Back on Y axis to clear 200 mm pallet size)
X0	Y0	Z0	D2000	AGV POSITION (15 mm below frame)
X0	Y0	Z200	D2000	200 mm above AGV POSITION

e.g. of a AGV PLACE routines

X0	Y0	Z200	D2000	200 mm above AGV POSITION
X0	Y0	Z 0	D2000	AGV POSITION (15 mm below frame)
X0	Y-230	Z0	D2000	Back on Y axis to clear 200 mm pallet size)
X0	Y-230	Z200	D2000	200 mm above AGV POSITION offset 230 mm in Y axis
X0	Y0	Z200	D2000	200 mm above AGV POSITION

In the LOCAL option, the commands are as follows: -

PICK ASRS 1 3 5	Picks a pallet from ASRS location SIDE 1, ROW 3, COLUMN 5
PLACE ASRS 2 3 5	Places a pallet in ASRS location SIDE 2, ROW 3, COLUMN 5
PICK CONVEYOR	Pick's a pallet from the Conveyor
PLACE CONVEYOR	Place's a pallet on the Conveyor
PICK AGV	Pick's a pallet from the AGV
PLACE AGV	Place's a pallet on the AGV

In the REMOTE option (i.e. CIM control), the commands are as follows: -

PICK KING PLACE CONVEYOR PICK CONVEYOR Picks up a King, places it on the conveyor then retracts the ASRS arm out of the conveyor

Denford RV-M1 Robot / Denford Linear Slide Start Up Procedure

Procedure to run the combined RV-M1 Robot / Denford / Denford Linear slide in the CIM System using the RS232 Robot Device Driver.

(Ensure that the RV-M1 Robot and Linear Slide Manuals have been read thoroughly and that all cable connections are made as per drawing supplied and all Linear Slide Positions have been taught)

Background.

The RV-M1 Robot Controller, controls the sequencing of Linear Slide movements in the CIM System by using a set of Generic Robot programs, these Robot programs utilise bit 0 of the Robots bottom 8 I/O bits as a **BUSY / IDLE** signal when used with the **RS232 Robot Device Driver**. The Robot Controller's top 8 I/O bits are used to communicate with the Linear Slide Controller. The Robot Controller receives commands via an **RS232 Robot Device Driver** Serial link from its associated Cell Controller Computer which acts in turn upon CIM software commands. Integrally in the Generic Robot programs are commands which output **Binary** signals in the correct sequence to the Linear Slide, these commands are interpreted by the Linear Slide Controller as **MOVE** command's. Note all commands for moves etc. both from the Linear Slide to the Robot and from the Robot to the System Interface Module are acknowledged with **BUSY** signals, which act as Interlocks, these **BUSY** Signals are usually the same Binary code as the original activating signals, but in the case of the Linear Slide the **BUSY** Signal is usually **Binary 1** (0001).

e.g. These command will move the Linear Slide to position No 6

350 OD &0601	0000 0110 0000 0001
360 OD &0E01	0000 1110 0000 0001
370 TI 2	Time delay
380 OD &0601	0000 0110 0000 0001

Initialising the Robot / Linear Slide combination for running in the CIM System.

(See the **General running** in the CIM Manual first)

1. Ensure that the CIM software is running and that the Green System Enabled and Red Emergency Stop Led's on the Right-hand front of the Interface Module are illuminated, these ensure the Emergency Stop circuit is active.

2. Ensure that both the Robot controller and the Linear Slide controller have mains supply connected to them of the correct rating. (The Robot controller front panel Yellow Led (Only when operation 4 below is carried out) and the Datum and Over travel sensors Led's on the Linear Slide should be illuminated.)

3. Ensure that the Linear Slide teach box is **Connected** to the controller and the Key switch is set to **REMOTE**.

4. Switch on the Robot controller. (The switch is at the upper Left on the rear of the controller)

5. If there is NO Beeping sound from the Robot controller then the Robot is OK, If beeping occurs check for illuminated Red Led's in the controllers right-hand side door panel, check the Led number and read off using the Robot manual for the cause of the problem. One of the common causes of the Beeping is the Emergency Stop circuit been broken or not initialised.

6. Ensure that the Robot is in a safe position, i.e. not inside a machine if the Emergency Stop operation has been previously activated.

7. Ensure that the Robot Teach Box is switched **OFF** and Toggle switch **ST1** in the robot controller **DOWN**.

8. Run the Device driver for the robot from the Cell / Host Computer and down load the NEST ROBOT & LINEAR SLIDE command.

9. Once the Robot has nested all its axes, it will then move to position 100 which is called the **PARK** position which is a **SAFE** position that the Robot always travels through when making different move sequences. After position 100 is reached the Linear Slide should then start to Datum, this consists of the Slide / Robot travelling towards the Datum sensor end of the Linear Slide, once the sensor detects the metallic button on the Slide belt the motor reverses so that the belt / metallic button moves away from the sensor about 20mm.

10. The Robot / Linear Slide combination are now ready to run in the CIM System.

Setting Linear Slide positions.

(See the **Linear Slide** Manual first)

1. Ensure that the Linear Slide teach box is **Connected** to the controller and the Key switch is set to **LOCAL**.
2. Ensure that the Robot is in a safe position and will not hit anything when the Linear Slide is homed.
3. Press the **RESET** button on the Teach Pendant and the Linear Slide will home.
4. After homing the Linear Slide, use the **JOG** and **SHIFT** buttons to move the Linear Slide carriage to the Left or Right to the desired position.
5. Set the thumb wheel to the correct position No required (Note position 0 should not be used because this is the home position).
6. Press the **SHIFT** and **POSITION** buttons together to register the position and associated position number.
7. Repeat the above for all other positions.

Procedure to load a RV-M1 Robot program from computer to controller using Windows "Notepad"and Windows "Terminal".

To run the RV-M1 Robot from WINDOWS 'TERMINAL' option proceed as follows :-

1. Create a TERMINAL program called 'RV-M1.TRM' by going into the 'SETTINGS' option and selecting 'COMMUNICATIONS'. Input all the relevant information i.e. :-

Baud rate	4800	
Data bits	7	
Stop bits	1	(Use 2 STOP BITS sometimes to get it to work)
Parity	None	
Flow Control	XON / XOFF	
COM	2	

Ensure STRIP LF is active

2. Turn ON the Robot teach box and press the button :- NST
   This is the Nesting button and will move all the robot axes back to their datum positions.

3. Using the Teach Box, move the robot axes to any arbitrary positions and press the buttons :-
   PS (Position Set)
   1 (Position Number)
   ENT (Enter)

   This as created the Position No 1. Repeat the above sequence and create Positions 2 and 3

4. Switch OFF the Robot Teach Box

5. Ensure both the switches in the RHS Robot Controller Panel are 'DOWN'.

6. Go into the WINDOWS 'NOTEPAD' option and create the following small test program

   DL 1,2048 (Deletes any old Programs in the Robot controller)
   10 RC 5 (Cycle 5 Times)

```
20 MO 1        (Move to Position No 1)
30 GC          (Gripper Closed)
40 MO 2        (Move to Position No 2)
50 GO          (Gripper Open)
60 MO 3        (Move to Position No 3)
70 NX          (Next)
80 ED          (End)
```

Save this program as 'TESTPROG.TXT'

7. Go into the WINDOWS 'TERMINAL' option and load the 'RV-M1.TRM' Program previously created. Select 'TRANSFERS' and then 'SEND TEXT FILE'. Now select 'TESTPROG.TXT' previously created and the Program created above will appear on the screen. Note ? (The 'TESTPROG.TXT can be copied from 'NOTEPAD' and Pasted into 'TERMINAL' RV-M1.TRM using standard Windows operations)

   Note ? The 'EDIT' then 'CLEAR BUFFER' command resets the Buffer and also clears the screen of figures.

8. TYPE the Command
   RS (RESET)

   This Resets the Robot Controller (i.e. The RED Error LED will go OUT)

9. TYPE the Command
   NT (NEST)

10. TYPE the Run the Program NEST)

    The Robot should Cycle through the Program 5 times.

11. To stop the Cycle press the 'START / STOP (TEST)' Button on the front of the Robot Controller. Pressing the 'RESET' Button on the front of the Robot Controller Resets the Robot Program back to its beginning.

Note? Because the XON / XOFF command was selected in 1. this could cause Program corruption when been Down Loaded due to the fact that when XON / XOFF is selected the robot Controller Buffer does not send back a Control Character to stop the flow of data. To overcome this problem if it occurs select 'SETTING' in the 'TERMINAL' Options with RV-M1.TRM

loaded and then select 'TEXT TRANSFER' and change the 'DELAY BETWEEN LINES' to a bigger value.

12. If you get NO response from the Robot i.e. No movement try typing the command
WH (WHERE)

If a line of figures appear on the screen as follows :-
+451, -432, -45 .....

Then the Serial communications link between the Computer and the robot Controller is OK, since it is transmitting the numerical position of each of the robot axis to the Computer.

APPENDIX A: Robot Program for Milling Cell

```
'*************************************
**
' DENMACH ROBOT PROGRAM (MILLER)
'
' 18 January 1993
' (C) Denford Machine Tools
'*************************************
**
' ROBOT POSITIONS
' PARK
' robot park          100
' MACHINE
' square to           11
' approach part       12
' almost around part  13
' around part         14
' retract             15
' PALLET
' square to           31
' approach part       32
' around part         33
' retract             35
'
'
'1 - Load the Miller from the Pallet
'2 - Release part in Miller and park
'3 - Move from park into Miller around part
'4 - Move from Miller to Pallet with part

'IF (USE MA) THEN
' vector 60mm in Z
PD 18,0,0,60,0,0
' vector 30mm in Z
PD 19,0,0,30,0,0
'ENDIF


DL 1,2048
 10 NT
'Initialise the robot
 15 GS 100
'The Main Loop
 20 OD 0
 30 ID
' Load Miller from pallet
 40 EQ 1,600
' Move into miller around part
 50 EQ 3,300
 60 GT 20

'*************************************
****
'Initialise the system...
'Move to park
100 MO 100, O
' Set the gripper pressure
110 GP 12,12,10
' Set the speed and acceleration
115 SP 9,H
150 RT

'*************************************
****
' COMMAND 2 - MOVE TO PALLET AND
GRAB PRIMARY PART
' Debounce input
600 TI 5
602 ID
604 NE 1,20
608 OD 1
' Park
612 MO 100, O
' Square up to pallet
624 MO 31, O
' Approach part
632 MO 32, O
636 SP 3,L
' Around part
640 MO 33, O
' Grip part
644 GC
646 TI 10
' Withdraw part
672 MO 35, C
676 SP 9,H
' Back to facing the conveyor
684 MO 31, C
'Now load the machine.
690 GT 224

'*************************************
****
```

```
' COMMAND 1 - PUT PART INTO MILLER
' Square up to miller
224 MO 11, C
'IF (USE MA) THEN
' Approach vice
232 MA 14, 18, C
236 SP 3,L
' Put part into vice
240 MA 14, 19, C
'ELSE
' Approach vice
'232 MO 12, C
'236 SP 3,L
' Put part into vice
'240 MS 13, 4, C
'ENDIF
242 MS 14, 4, C
243 GO
' Grip part
' COMMAND 2 - LEAVE MILLER AND
PARK
' Interlock until chuck is closed
244 OD 0
248 ID
252 NE 2, 248
' Debounce input
256 TI 5
257 ID
260 NE 2, 248
262 OD 2
' Open gripper
264 GO
266 TI 10
268 SP 3,L
' Move away from part
272 MO 15, O
' Out of machine
276 SP 9,H
' Back to facing the machine
284 MO 11, O
' Move to park
288 MO 100, O
' Finished so send idle signal
292 OD 0
' Return to main loop
296 GT 20

'**********************************
****
' COMMAND 6 - MOVE INTO MILLER AND
GRAB PART
' Debounce input
300 ID
304 NE 3,20
308 OD 3
' Park
312 MO 100, O
320 SP 9,H
' Square up to machine
324 MO 11, O
' Approach part
332 MO 15, O
336 SP 3,L
' Around part
340 MO 14, O
346 TI 5
' COMMAND 4 - TAKE PART FROM
MILLER AND PLACE IN PALLET
' Interlock until PartHolder is open
348 OD 0
'loop.
352 ID
356 NE 4, 352
' Debounce input
360 TI 5
362 ID
364 NE 4, 352
366 OD 4
' Grip part
369 GC
' Withdraw part
370 TI 10
'IF (USE MA) THEN
372 MA 14, 19, C
374 MA 14, 18, C
'ELSE
'372 MS 13, 4, C
'374 MS 12, 4, C
'ENDIF
376 SP 9,H
' Back to facing the machine
384 MO 11, C
' Now put the part back in the pallet.

' Square up to conveyor
424 MO 31, C
```

```
' Approach pallet
432 MO 35, C
436 SP 3,L
' Put part into pallet
440 MO 33, C
' Open gripper
464 GO
466 TI 10
468 SP 3,L
' Move away from part
472 MO 32, O
476 SP 9,H
' Back to facing the conveyor
484 MO 31, O
' Move to park
488 MO 100, O
' Finished so send idle signal
492 OD 0
496 GT 20
```