The Production Cell
A Real-Time Case Study

A P Robson & W D Henderson & M J Brockway
Computing, University of Northumbria

January 22, 2013

1 Overview

The Production Cell has two robots, a conveyor and a control panel. Figure 1 shows the physical layout of the cell. At its simplest level, items are picked up by one robot and placed on the conveyor. They pass along the conveyor and are picked off at the delivery end by a second robot and placed on an output pad. The control panel allows an operator to supervise the system.

There are a number of sensors positioned to detect items as they pass through the cell. The sensors are associated with the various components to form subsystems; and each subsystem is managed by a micro-controller. The micro-controllers are connected by a communications network to make the complete system. The micro-controllers use the network to coordinate their actions and move items through the Production Cell. Figure 2 gives a schematic overview of the system.

1.1 Subsystems

There are four systems components:

Input robot Picks a work-piece up from the load platform and places it on the conveyor.

Conveyor Transports the work-piece to the unload position. It has two sensors, one at each end of the conveyor belt.

Output robot Removes a work-piece from the conveyor and places it on the unload platform.

Control Panel Allows the operator to enter control commands, and manages load and deposit platform sensors. It is responsible for the load and unload platform sensors.

The solution is required to manage each of these components with independent software subsystems. These subsystems should be implemented as distributed concurrent tasks communicating with messages. Note that the subsystems must synchronise with each other. For example, a protocol could be established between *Robot Manager* and *Conveyor Manager* with the events *load*, *readyToLoad* and *loaded* such that items can be safely put onto the conveyor by the robot.

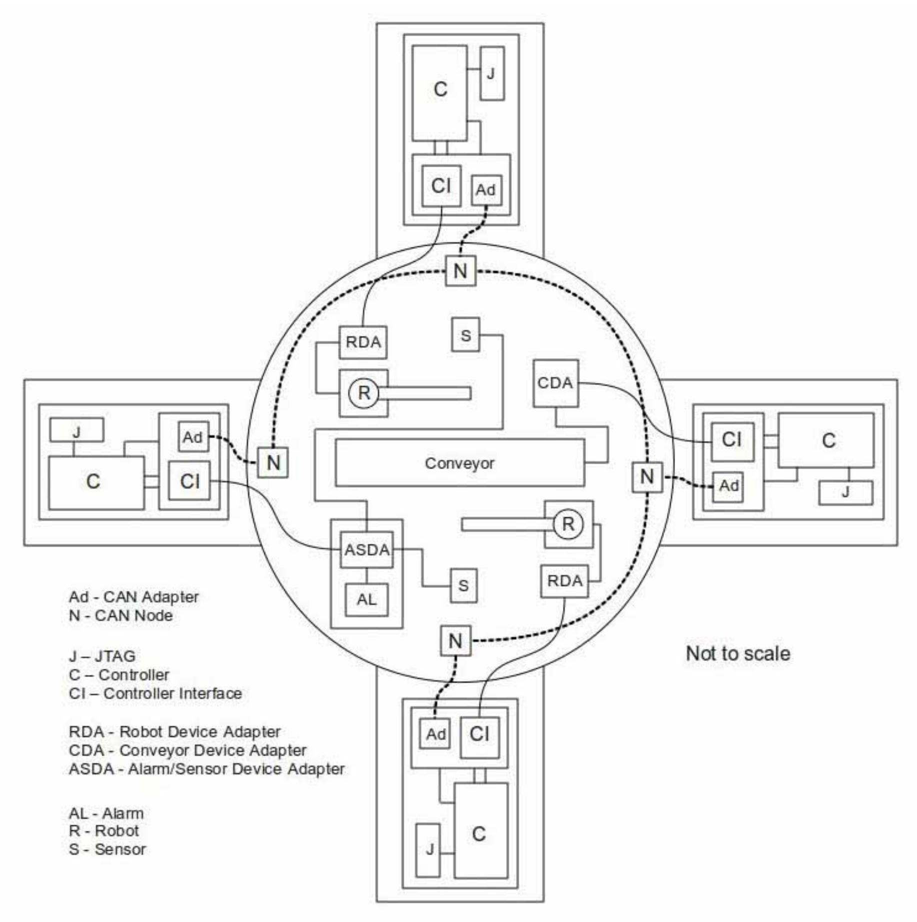

Figure 1: Production Cell Physical Arrangement

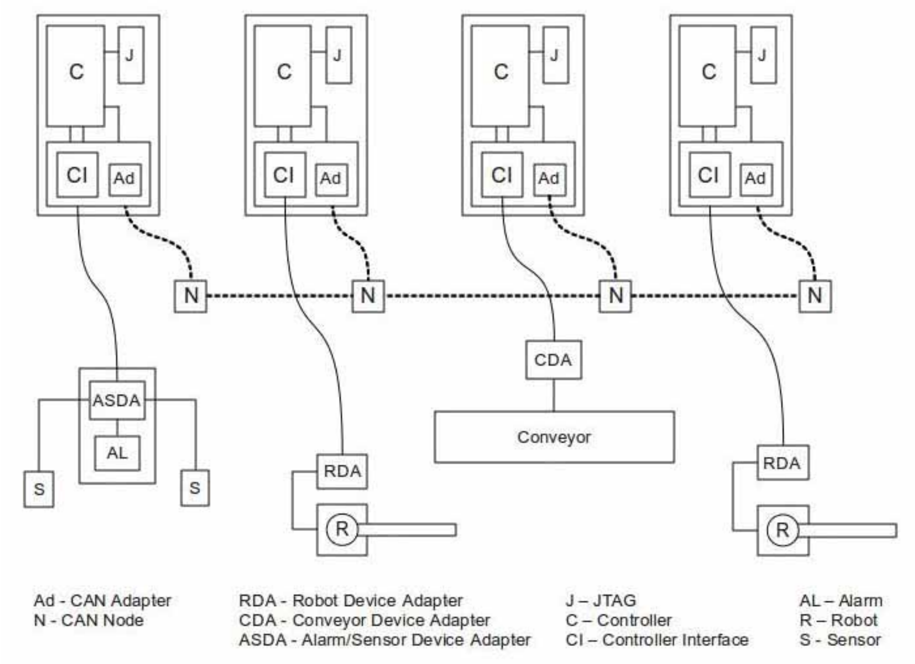

Figure 2: Production Cell Schematic

2 Modes of Operation

The system can be characterised by the states: ready, running, paused and error.

2.1 Powering up the System

When the system is powered up (and its control software loaded) it is ready but not running. There must be no work-pieces in the Production Cell or on its load and unload platforms when it is powering up.

2.2 Starting and Stopping the System

Starting the system makes it able to receive work-pieces as described in §2.3. There must be no work-pieces in the Production Cell when it is started, but there can be one on the load platform. After the system is started, it is considered to be ready.

Once the system is started it can be stopped, which closes it down in a controlled manner. When a stop is initiated, the system waits for any in-progress work-pieces to leave the Production Cell, and then shuts down all of its subsystems. A work-piece waiting on the load platform should be left there.

2.3 Normal Operation

A workpiece enters the Production Cell by being placed on the the load platform. It is picked up by the input robot and placed on the conveyor. The piece is transported to the conveyor's far end, where it is picked up by the output robot and deposited on the unload platform. The conveyor must be stationary during pickup and deposit operations. The item leaves the Production Cell by being removed from the unload platform. Once a work-piece has been placed on the load platform, it can only be removed by the input robot.

Multiple work-pieces are allowed in the Production Cell at the same time, and the subsystems can run concurrently. So work pieces can be simultaneously handled by both robots and the conveyor, and the conveyor can hold more than one work-piece.

2.4 Pause Operation

The system can be paused when it is in normal operation mode. When the system is paused it should stop processing work-pieces as soon as possible; but the subsystems can finish partially complete operations. When the system is resumed it should immediately recommence processing.

Work-pieces can be placed on the load platform and removed from the unload platform while the system is paused. Otherwise the physical state of the Production cell will not be changed while it is paused.

2.5 Error Operation

All subsystems immediately shut down when an error occurs. An error is a serious fault that is irrecoverable, or one that could cause damage to the Production Cell or its operators. Errors can be automatically generated by any subsystem or be manually raised at the control panel. The system is not ready after an error has happened.

The system must be reset to recover for an error, which puts it into stopped mode as though it has just been powered up (see §2.1). Any work-pieces in the Production Cell must be removed before it is reset.

3 Component Function

3.1 Control Panel

The control panel consists of four push buttons, four lights, and a small LCD. These are partly on the ARM controller board and its associated interface PCB, and partly (an emergency stop button and alarm indicator) on a separate circuit board. The buttons are Start/Stop, Pause/Resume, Error and Reset; and the lights are Ready, Running, Paused and Emergency.

The control panel subsystem is also responsible for sensors on the load and unload platforms.

3.1.1 Start/Stop Buttons and Running Light

Pressing Start/Stop starts the system if it is stopped, and stops the system if it is started. When the system is stopped the running light is off, and when the system is running the running light is on. After the Start/Stop button is pressed the running light flashes until the system is completely started or stopped as required. It is an error if start or stop does not complete as required. (See §2.2 for more information.)

3.1.2 Pause Button and Pause Light

Pressing Pause/Resume puts the system into a paused condition if it is not paused, and removes the paused condition if it is paused. The pause light is on if and only if the system is paused. The pause light flashes while the system is in transition to or from a paused state. (See §2.4 for more information.)

3.1.3 Reset Button and Ready Light

Pressing the Reset Button causes the system to go into its 'just powered up' state if it is in an error state. (See §2.1 for more information.)

The Ready Light is on when all components are ready and there is no error, The Ready Light flashes while the system is waiting for all of its components to confirm their ready status. It is an error condition if pressing reset does not successful reset all components.

3.1.4 Emergency Button and Emergency Light

The Emergency Light is switched on if there is a error generated by pressing the Emergency Button or if an error is generated by another subsystems. Pressing the Emergency Button reports an error to the rest of the system. The emergency condition is clear by the Reset Button. (See §3.1.3 for more information.)

After an emergency stop or any other unrecoverable error the system is not ready and the Ready Light is turned off. All components are informed of emergency conditions. (See §2.5 for more information.)

3.1.5 Liquid Crystal Display

The role of the LCD is not defined. It can be used to show the system's current status, and for error and diagnostic messages.

3.1.6 Sensors

The load sensor detects the arrival of a new work-piece at the Production Cell, and the unload sensor detects that a work piece is available for removal.

3.2 Robots

The robots have three joints: waist, shoulder and elbow that can be independently controlled to move a jawed gripper in three dimensions. The gripper can be opened and closed. The robot has no feedback, in particular it cannot tell if it has picked up a work-piece.

There are two robots: one for the insertion and another for extraction of work-pieces. Their general operations have the same pattern. Typically a robot:

1. Waits for a request to pick up an work-piece;

2. moves the object to the 'put down wait' position;

3. asks permission to deposit;

4. deposits the item;

5. and finally returns to its ‘wait for request’ position.

These movements are interleaved with communications to other cell subsystems to coordinate and verify operation.

If the robot fails to successfully pick up a piece, it should retry the operation. Successive failures should be treated as a non-recoverable fault and reported as an error. A failure to deposit is also a non-recoverable fault.

The two robots will share some common functionality and this should be exploited where possible. However, the operational functionality of the two robots in terms of communication and synchronisation requirements will be quite different.

3.3 Conveyor

The conveyor has two controls: on/off and forwards/backwards; and it has two sensors. Work-pieces are place on and removed from the conveyor by the robots. The conveyor should be stopped when a work-piece is being deposited or picked up.

Items are carried down the conveyor passing sensors located near to each end. Thus, it is possible to detect if work-pieces have been successfully loaded onto and removed from the conveyor.

The system should be as responsive as possible with the conveyor acting as a storage buffer of work-pieces between the two robots. This implies that more than one item can be on the conveyor at the same time.

4 Non-Functional Requirements

There are a number of non-functional requirements, as follows:

- The system's control must be distributed, with each subsystem being as autonomous and asynchronous as possible. In particular, the design should not have a client-server architecture, or use the control panel's micro-controller as a central manager.

- The solution must be written in ANSI C. A safe subset is not required.

- The software must use the supplied Applications Program Interface, which supports all of the system's components.

- Each of the four subsystems must be developed, tested and be capable of being demonstrated separately if required. This will require simple test environments for each component, which simulate the communication of messages to and from other components.

- The robot systems must store position coordinates in data structures that can be easily replaced. It is recommended that they are incorporated into the software as independent compilation units.

- Implement subsystem communication and synchronisation using messages.
- A real-time operating system can be used, but is not required. The system does not have to be homogeneous: The designer of each subsystem can make their own decision.
- You may find it useful to report the status of system components using the on-board LEDs.

5 A Physical Implementation

A small scale Production Cell has been built as part of the School's Real-Time Laboratory [7]. It uses ARM7 LPC2378 micro-controllers on NXP LPC2378-STK evaluation boards [3][4][5], and its communications network is CAN [1].

The IAR Kickstart integrated development environment [6] is used for programming and downloading, editing and build management. The μC/OS operating system [2] is used if required.

References

[1] *CAN Specification Version 2.0*, 1991, Robert Bosch GmbH, 1991.

[2] Labrosse, J.J., *MicroC/OS II: The Real Time Kernel.* Osborne McGraw-Hill, 2002.

[3] *LPC2377/78 Product Data Sheet*, rev 04 NXP Semiconductors, 2008.

[4] *LPC23XX User Manual*, rev 01 NXP Semiconductors, 2008.

[5] *Introduction to the LPC2000* Hitex Ltd, 2006; ISBN: 0-9549988 1.

[6] *IAR Embedded Workbench IDE User Guide*, 17th ed IAR Systems July 2010

[7] Adrian Robson & William Henderson, *Real Time Laboratory Mk II Technical Guide*, Version 4, Northumbria University, Feb 2010.