A unified framework for pull control mechanisms in multi-stage manufacturing systems

George Liberopoulos [a] and Yves Dallery [b]

[a] *Department of Mechanical and Industrial Engineering, University of Thessaly, Pedion Areos, GR-38334 Volos, Greece*

[b] *Laboratoire Productique-Logistique, Ecole Centrale de Paris, Grande Voie des Vignes, F-92295 Chatenay-Malabry, France*

This paper presents a unified framework for pull production control mechanisms in multi-stage manufacturing systems. A pull production control mechanism in a multi-stage manufacturing system is a mechanism that coordinates the release of parts into each stage of the system with the arrival of customer demands for final products. Four basic pull production control mechanisms are presented: Base Stock, Kanban, Generalized Kanban, and Extended Kanban. It is argued that on top of any of these basic coordination mechanisms, a local mechanism to control the work-in-process in each stage may be superimposed. Several cases of basic stage coordination mechanisms with stage work-in-process control are presented, and several production control systems that have appeared in the literature are shown to be equivalent to some of these cases.

Keywords: pull production control, kanban, base stock

1. Introduction

Manufacturing systems consist of machines and workstations where operations such as machining, forming, assembly, inspection, testing, etc., are carried out on raw-material parts, fabricated components, and sub-assemblies to create final products to be delivered to customers.

The effective production control in any manufacturing system, that is, the management of the total flow of goods through the system, from the acquisition of raw parts to the delivery of final products to customers, is key to the competitiveness of the system. Production control is an optimization problem that typically addresses the question of when and how much to produce in order to achieve a satisfactory customer service level (measured by how quickly customer demands are satisfied), while keeping low in-process inventories. Difficulties in production control arise because of queueing delays due to variability in production capacity (e.g., due to the failure or maintenance of a machine) and demand for final or intermediate products.

One approach to tackling the production control is to formulate it as an optimal control problem and then try to determine an optimal control policy for this problem (e.g., see [13]). Thus far, this approach has been successful only for very simple

systems. Moreover, an optimal policy, assuming one can be found even for realistic systems, risks being too complicated to be of any practical value. Optimal control analysis, however, is still valuable in that any information on the optimal policy or its structure that it may reveal, even for small-sized problems, may point to the design and help to assess the performance of good heuristic policies for more complex systems.

A more practical approach to tackling the production control problem is to restrict the search for a production control policy to a class of simple, sub-optimal policies that are easy to implement and try to determine the optimal policy within this class. Much of the research effort in this area has focused on developing and evaluating simple production control policies that depend on a small number of parameters and have often emerged from actual industrial practice. In many of these policies production is triggered by actual demands for final products. Such policies are often referred to as *pull* production control policies or systems or mechanisms (in this paper, the words "policy", "system", and "mechanism" are used interchangeably to mean the same thing).

In many pull control systems encountered in the literature production control is applied at a selected number of points in the manufacturing system. This is done by functionally aggregating several production activities into different production stages or cells and then coordinating the release of parts into each stage with the arrival of customer demands for final products. Some systems, in addition to stage coordination, also impose local control on the work-in-process (WIP) within each stage, which implicitly or explicitly also affects the release of parts from one stage to the other.

Some of the names of pull production control systems that have appeared in the literature are: *Base Stock* (e.g., [7]), *Hedging Point* [13], *Kanban* (e.g., [2]), *CONWIP* [15,20], *Generalized Kanban* [5] *Extended Kanban* [10,11], *Local Control* [7], and *Integral Control* [7].

Some pull production control systems have appeared in the literature as blocking mechanisms in the more restrictive case of a multi-stage manufacturing system where each stage consists of a single machine that is prone to blocking. Some of the names of such mechanisms are: *Manufacturing Blocking* (e.g., [18]), *Minimal Blocking* [18], *Generalized Blocking* [8], *CONWIP/Kanban Hybrid* [3], *Base Stock/Kanban Hybrid* [3], and *Manufacturing Blocking/Hedging Point Hybrid* [14].

Although much work has been done on individual pull production control systems, few comparison studies exist. This is partly due to the fact that different systems have been described within different frameworks. A common framework describing and modeling different approaches was developed in [6] and [7]. There, a system for coordinating and controlling the material and part flow within a multi-stage system, called the PAC (Production Authorization Card) system, was introduced. It was then demonstrated how, through the appropriate choice of parameters, the PAC system can be specialized into a wide variety of classical coordination approaches. The PAC system is a general system that includes batching of parts and time-delays to deal with advanced information on the demand. A PAC system with unit batch sizes and zero time-delays reduces to the Generalized Kanban Control System that we describe in

section 3.2.1. The PAC system, as general as it may be, does not include all other mechanisms as special cases. For example, it does not include the Extended Kanban Control System (see [10,11] and section 3.2.2) and Gershwin's scheme (see [14] and section 6.4). Moreover, presenting a mechanism as a special case of the PAC system (or any other general system for that matter) may not always be the best way of explaining how this mechanism works. For example, it is difficult to see how the CONWIP system (see [15,20] and section 3.1.2) works when it is seen as a special case of the PAC system with a single machine per stage as is proposed in Buzacott and Shanthikumar's book [7].

This paper presents a unified framework for pull production control mechanisms in multi-stage manufacturing systems and shows how these mechanisms are related to each other. Initially, four basic pull control mechanisms are presented: Base Stock, Kanban, Generalized Kanban, and Extended Kanban. It is argued that on top of any of these basic coordination schemes, a local mechanism to control the WIP in each stage may be superimposed. Several cases of basic stage coordination mechanisms with stage-WIP-control are presented, and several production control systems that have appeared in the literature are shown to be equivalent to some of these cases. Finally, it is argued that it possible to nest several pull production control systems to create new systems. This is done by specifying a hierarchy of stages where higher level stages are formed by aggregating lower-level stages and a different pull production control mechanism is used at each stage level.

The remainder of this paper is organized as follows. In section 2 we present the nomenclature that will be used to model the basic pull production mechanisms presented in section 3. In section 3 we present four basic pull production control systems, each representing a different philosophy on how demands and production authorizations are propagated throughout the system and used for stage coordination. These systems are the Base Stock, Kanban, Generalized Kanban, and Extended Kanban. In section 4 it is argued that on top of any of the basic pull control mechanisms for stage coordination it is always possible to add a local WIP-control mechanism within each stage. Three WIP-control mechanisms are presented. In section 5 several pull control systems that combine basic pull control systems with local WIP-control are presented. In section 6 we present pull control mechanisms on single-machine-per-stage manufacturing systems and show the equivalence of these mechanisms to various blocking mechanisms. Section 7 discusses how more complex pull control mechanisms can be obtained by nesting simple pull control schemes. Conclusions are drawn in section 8.

2. Modeling definitions on pull production control systems

We consider a manufacturing system in which several production activities have been functionally aggregated into different production stages so that pull production control can be exercised in between stages, coordinating the release of parts into each stage with the arrival of customer demands for final products.

There are many reasons for wanting to aggregate production activities into stages and control the material flow in between stages. First, in most manufacturing systems production activities are naturally grouped into well identifiable production stages. In the Semiconductor Manufacturing Industry, for example, the following stages are well identifiable: circuit design and mask preparation, wafer preparation, wafer fabrication, probe test and sort, assembly, and test and classify. In practice, these stages operate independently of one another and what couples them is the release of parts from one stage to the next. Second, when dealing with multi-product systems, setups to change from one product to another are often performed on whole sub-systems of machines (e.g., on a production line) rather than on individual machines. Controlling the production of each individual machine may, therefore, not be appropriate in such cases. Finally, having fewer points to control makes the production control problem simpler and the implementation of a production control policy easier.

It should be pointed out that the aggregation of production activities into stages considered here is purely functional or logical and that it is done for production control purposes only. By this we mean that a physical manufacturing system may be aggregated into stages for production control purposes in many different ways. Two extreme cases are: (1) the case where the entire physical manufacturing system is aggregated into a single-stage and (2) the case where each machine of the physical manufacturing system belongs to a different stage. In the first case, production control is exercised at the entry of the manufacturing system, deciding when to release a part into the system. In the second case, production control is exercised at the machine level, deciding when to release a part for production at each machine.

Because of the range of complexity of possible system structures and coordination policies, we will focus on the special case of a manufacturing system that consists of several production stages in series, produces a single part type, and does not involve batching, reworking or scrapping of parts, and time-delays to deal with advanced information on the demand. The ideas presented in this paper, however, could be extended to more complex systems.

Each stage is a production/inventory system made of an *input buffer*, a *manufacturing facility*, and *an output buffer*. The manufacturing facility at each stage is a sub-system of the original manufacturing system, containing one or more production activities (e.g., a single machine, a production line, a job shop, a flexible manufacturing cell, etc.). For ease of exposition, all our illustrations will be on systems with two stages in series, but everything we present applies to the general case of N stages in series. Figure 1 shows a manufacturing system with two stages in series. The manufacturing facility at each stage is drawn as an oval and the input and output buffers are drawn as "∇"s.

The problem we are concerned with is how to coordinate the release of parts into each stage and the arrival of customer demands for final products at the end of the system.

In section 3 that follows we present four basic pull production control systems, the Base Stock, Kanban, Generalized Kanban, and Extended Kanban Control Systems.

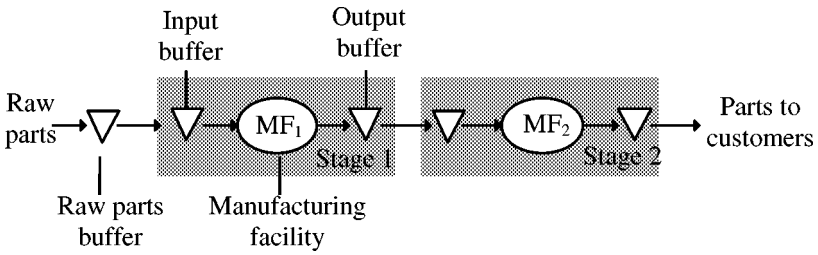

Figure 1. A manufacturing system with two stages in series.

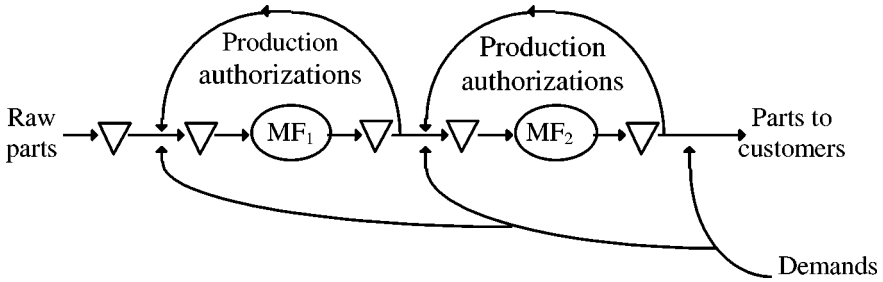

Figure 2. Flow of parts, demands, and production authorizations in a manufacturing system with two stages in series.

These systems represent different ways of stage coordination, but they all have the following common characteristics.

Every basic pull control system has three types of moving elements: *parts, demands*, and *production authorizations* (actually, the Base Stock Control System presented in section 3.1.1 does not use any production authorizations at all, but it can also be viewed as using an infinite number of production authorizations).

In every basic pull control system, in order for a part to be released from the output buffer of Stage $i-1$ into the input buffer of Stage i, the following conditions must be met:

1. There must be at least one finished part in the output buffer of Stage $i-1$.
2. There must be at least one demand to release a new part into Stage i.
3. There must be at least one production authorization to release a new part into Stage i.

The timing when parts, demands, and production authorizations move from one area to the other in the manufacturing system depends on the pull control system in place. The general principles of how these elements move, however, are the same in all pull control systems. More specifically, parts, demands, and production authorizations move from one area of the manufacturing system to the other as follows (see figure 2).

A part begins its trajectory from the raw-materials buffer, moves downstream the manufacturing system from one manufacturing stage to the next, and exits the

system to be delivered to a customer. More specifically, a part is released from the output buffer of a stage into the input buffer of the downstream stage when the three conditions above are met. Once a part is released into the input store of a stage, it moves on to the manufacturing facility of that stage, as soon as possible (e.g., as soon as the first machine is available). There, it receives processing on the machines. Once a part has completed processing in the manufacturing facility of a stage, it is stored in the output store of that stage where it remains until it is released into the input store of the downstream stage.

A demand works its way upstream the manufacturing system in the following sense. When an independent customer demand for a final product arrives at the manufacturing system, it generates a dependent demand for the release of a part into each stage. These dependent demands are transmitted to their respective stages one by one starting from the last stage. The timing when these demands are transmitted to their respective stages depends on the pull control system in place. When a demand is satisfied, it is dropped from the system.

A production authorization is associated with a particular stage and traces a closed path through that stage. Each stage contains a fixed number of production authorizations. Initially, a production authorization waits at the entrance of the stage to authorize the release of a part into that stage. When a part is released into the input buffer of the stage, the production authorization is attached onto the part and follows it through the stage. The production authorization is liberated from the part before the part is released into the next stage. At some point, depending on the pull control mechanism in place, the liberated production authorization returns to the beginning of the stage waiting to authorize the release of a new part into the stage. A production authorization is a functional element of the control system. In practice, a production authorization of a stage may be a physical card or a fixture associated with that stage, as is the case of a kanban, or it may be a logical flag in a production control software package.

It should be noted that the manufacturing facilities and input and output buffers are logical areas that may or may not correspond to distinct physical areas in the real manufacturing system. The transfer of a part from one area to another, therefore, is also logical and not necessarily physical. For instance, the output buffer of a stage and the input buffer of the downstream stage may correspond to the same physical warehouse. In this case the logical transfer of a part from one buffer to the other may not correspond to a physical transfer of the part but just to a change in its status. Also, it should be noted that the manufacturing facilities may actually represent transportation rather than production activities.

There are several ways to model pull control systems. In this paper, pull production systems are modeled as queueing networks with *synchronization stations*. A synchronization station consists of a server with instant service times. The server is fed by two or more queues and operates as follows. As soon as there is at least one customer in each of the queues that feed the server, these customers move forward into the server. This implies that at any time at least one of the queues that feed the server is empty.

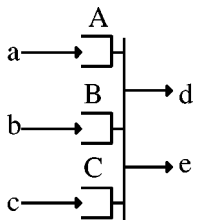

Figure 3. A synchronization station with three input queues and two outgoing customers.

Customers that enter the server, immediately exit the server after possibly having been separated into more or joined into fewer customers. In the pull control systems we present in the following sections, the queues in a synchronization station may contain parts, demands, production authorization cards, or combinations of the above.

An example of a synchronization station with three input queues and two outgoing customers is shown in figure 3. To see how this synchronization operates, suppose that initially Queues A and B contain four and three customers, respectively and that Queue C is empty. Then, as soon as a customer arrives to Queue C, three customers, one from each queue, are merged into two customers, d and e, and exit the synchronization station. After this incident, the contents of Queues A, B, and C are 3, 2, and 0, respectively.

All queues in the queueing network models are assumed to have infinite capacity. This does not necessarily mean that the physical spaces they represent, if any, are infinite. Finite spaces are modeled with the use of closed queueing sub-networks in which a fixed number of tokens circulate. Free tokens indicate free spaces whereas engaged tokens represent occupied spaces.

With these modeling definitions in mind, we present four basic pull control systems in the following section.

3. Basic pull control systems

In this section we present four important pull production control systems that we consider as forming the basis of other systems. These systems are the Base Stock, Kanban, Generalized Kanban, and Extended Kanban Control Systems. The first two systems depend on one parameter per stage, whereas the last two systems depend on two parameters per stage. Although each two-parameter-per-stage system includes both single-parameter-per-stage systems as special cases, we present the single-parameter-per-stage systems first, because we want to emphasize that these systems are the basic building blocks of the two-parameter-per-stage systems rather than special cases of them.

All systems are modeled as queueing networks with synchronization stations. The names of the queues in these networks imply their contents and are common in all systems. Thus, in any of these queueing network models, Queue P_i contains Stage-i finished parts, Queue D_i contains demands for the production of new Stage-i

finished parts, Queue A_i contains Stage-i production authorizations, Queue PA_i contains pairs of Stage-i finished parts and Stage-i production authorizations, and Queue DA_i contains pairs of demands for the production of new Stage-i finished parts and Stage-i production authorizations, for $i = 1, \ldots, N$, where N is the number of stages. Queue I_i, $i = 1, \ldots, N$, contains Stage-i parts that are waiting to be processed in the manufacturing facility of Stage i. In many models Queues I_i play no role in production control but are drawn anyway for consistency. Finally, Queue P_0 is the raw-parts buffer, and Queue D_{N+1} contains demands for the delivery of finished products to customers. Raw parts are assumed to arrive to the raw-parts buffer according to a given arrival process that is outside the scope of production control.

3.1. Single-parameter-per-stage pull control systems

Two fundamental pull control systems are the Base Stock and the Kanban Control Systems. Each system depends on one parameter per stage and represents a diametrically different philosophy on how demand flows through the system and is used for production control. The Base Stock Control System is presented in section 3.1.1 and the Kanban Control System is presented in section 3.1.2.

It should be pointed out that apart from these two basic, single-parameter-per-stage systems, there exist other pull production control systems that depend on one parameter per stage. One such system is the Integral Control System presented in appendix A. It uses elements of both the Base Stock and the Kanban Control Systems.

3.1.1. Base Stock Control System (BSCS)

One of the simplest and most well-known pull control systems encountered in the literature is the *Base Stock Control System* (BSCS). Figure 4 shows the queueing network model of a BSCS with 2 stages in series. Queue P_i represents the output buffer of Stage i, $i = 1, 2$. Queue D_i contains demands for the production of new Stage-i finished parts, $i = 1, 2$. Queue P_0 represents the raw-parts buffer, and Queue D_3 contains customer demands. Queue I_i represents the input buffer of Stage i, $i = 1, 2$.

When the system is in its initial state, that is, before any demands have arrived to the system, P_0 contains an initial number of raw parts, P_i contains S_i Stage-i finished parts, $i = 1, 2$, and all other queues in the system are empty. S_i is the only control parameter of Stage i and is referred to as the base stock of Stage i, where the term

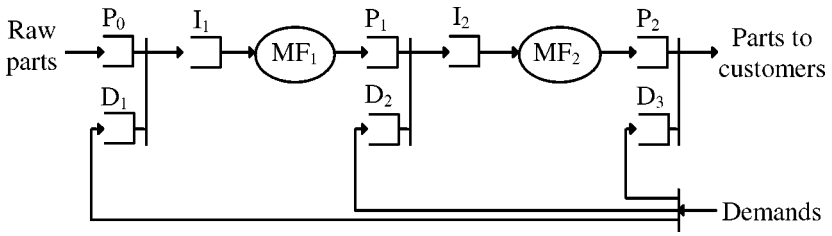

Figure 4. A BSCS with two stages in series.

base stock, also known as *installation stock* [1], is borrowed from inventory systems theory.

In the BSCS there are no production authorizations. All that is needed for a part to be released from the output buffer of a stage into the input buffer of the next stage is a demand for the release of such a part. Alternatively, the BSCS can be viewed as a pull production system that has an infinite number of production authorizations in every stage.

The BSCS operates as follows. When a customer demand arrives to the system, it joins Queue D_3 thereby requesting the release of a finished part from P_2 to the customer. The customer demand instantly also generates a demand in D_i, $i = 1, 2$, that authorizes the release of a part from P_{i-1} to I_i, $i = 1, 2$.

The philosophy of the BSCS is the following. When a customer demand arrives to the system, it is immediately transmitted to every stage in the system, authorizing it to immediately start working on a new part, which it pulls from the output buffer of its upstream stage, provided that such a part exists.

The advantage of this mechanism is that it responds rapidly to demand. Its disadvantage is that it provides a very loose coordination between stages and that it does not guarantee any limit on the number of parts that may enter the system, since every demand arriving to the system authorizes the release of a new raw part into the first stage. A way to overcome this disadvantage is to impose an additional control on the WIP in each stage, as we will see in section 4.

Equivalent system to the BSCS: Hedging Point Control System (HPCS)

It is worth noting that the BSCS is equivalent to the so-called *Hedging Point Control System*. The Hedging Point Control System (HPCS) has its origins in Kimemia's and Gershwin's optimal control approach to the production flow control problem (see [13,17]). In the HPCS, the policy to release a new part into a stage depends on the difference between the cumulative number of parts that have already been released for production into that stage and the cumulative number of customer demands that have arrived to the system. This difference corresponds to inventory, when positive, and backlog, when negative, and is referred to as the *inventory/backlog position* of the stage [13]. The HPCS authorizes the release of a part into Stage i if the inventory/backlog position of that stage is below a given, non-negative level called *hedging point* [13] or *echelon stock* [1] and denoted Z_i. The idea is to drive the inventory/backlog position of every stage towards its hedging point at times of excess capacity in order to hedge against future capacity shortages. The hedging points are non-increasing as stages increase, that is, $Z_i \geqslant Z_{i+1}$, for all i. Theorem 1 states that the HPCS is equivalent to the BSCS.

Theorem 1. An N-stage HPCS with hedging points Z_i, $i = 1, 2, \ldots, N$, is equivalent to a BSCS with parameters

$$S_N = Z_N,$$

$$S_{N-1} = Z_{N-1} - Z_N,$$
$$\ldots$$
$$S_1 = Z_1 - Z_2.$$

The proof of theorem 1 is in appendix B. For the special case where $N = 2$, theorem 1 states that a two-stage HPCS with hedging points Z_1 and Z_2 is equivalent to a BSCS with parameters $S_2 = Z_2$ and $S_1 = Z_1 - Z_2$.

3.1.2. Kanban Control System

By far the most popular pull control system is the *Kanban Control System* (KCS). The KCS was first implemented in the Toyota production line in the mid-seventies and is often used to exemplify just-in-time production. The word *kanban* means "card" or "tag" in Japanese and refers to the mechanism whereby a production authorization card is attached onto a part authorizing its release into a stage. The last two decades have seen a surge in the literature on the KCS, but there seems to be no agreed-upon definition on what a KCS is. A recent review describing alternative KCS definitions is [2]. A comparison of alternative kanban control mechanisms is given in [19]. Our definition of Kanban control coincides with that of [7]. It is the most general definition in the sense that it applies to an arbitrary, multi-stage manufacturing control system, whereas other definitions are restricted to single-machine-per-stage control systems. Even in this particular case, these other definitions are variations of the definition considered in this section (see section 6.2).

Figure 5 shows the queueing network model of a KCS with two stages in series. Stage i, $i = 1, 2$, has associated with it K_i production authorizations or kanbans. Queue PA_i represents the output buffer of Stage i and contains pairs of Stage-i finished parts and production authorizations, $i = 1, 2$. Queue DA_i contains pairs of demands and production authorizations for the production of new Stage-i finished parts, $i = 1, 2$. Queue P_0 represents the raw parts buffer, and Queue D_3 contains customer demands. Queue I_i represents the input buffer of Stage i, $i = 1, 2$.

When the system is in its initial state, that is, before any demands have arrived to the system, P_0 contains an initial number of raw parts, PA_i contains K_i

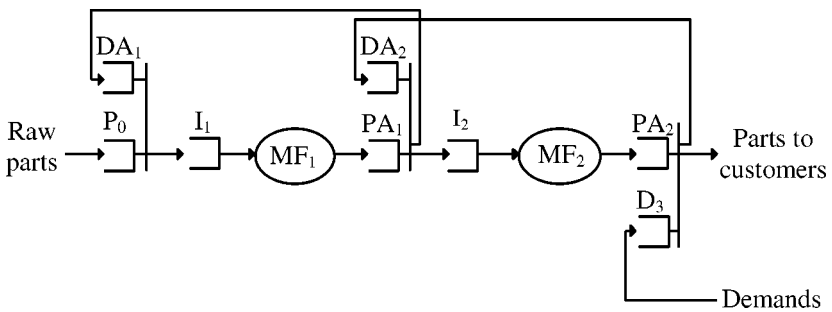

Figure 5. A KCS with two stages in series.

Stage-i finished parts, each part having a Stage-i kanban attached to it, $i = 1, 2$, and all other queues in the system are empty. K_i is the only control parameter of Stage i.

The KCS operates as follows. When a customer demand arrives to the system, it joins Queue D_3, thereby requesting the release of a finished part from PA_2 to the customer. If a part is available in PA_2, it is released to the customer after liberating the kanban that was attached to it. This kanban is transferred upstream to DA_2 carrying along with it a demand for the production of a new Stage-2 finished part and authorizing the release of a finished part from PA_1 into I_2. If a part is available in PA_1, it is released into Stage 2 after liberating the Stage-1 kanban that was attached to it and engaging the Stage-2 kanban in DA_2. The liberated Stage-1 kanban is transferred upstream to DA_1 carrying along with it a demand for the production of a new Stage-1 finished part and authorizing the release of a finished part from P_0 into I_1. This way the customer demand that originally arrived to D_3 is transferred upstream "riding" on the kanbans. If at some Stage i a finished part is not available in PA_i, no kanban is transferred upstream and the transfer of the customer demand is put to a halt; it is resumed when a part becomes available in PA_i.

The philosophy of the KCS is that a customer demand is transmitted upstream of the system from Stage i only when a finished part is released downstream of Stage i.

The KCS provides tighter coordination between stages than does the BSCS. In the KCS, a stage is authorized to start working on a new part exactly when it has released a finished part to the next stage. In the BSCS, on the other hand, a stage is authorized to start working on a new part when a customer demand for a final product arrives to the system. An advantageous consequence of the operation of the KCS is that the number of parts in Stage i is limited by the number of Stage-i kanbans. A disadvantage is that the system may not immediately respond to demand, since a customer demand may not immediately be transferred to all stages upon its arrival to the system. Another drawback is that the transfer of demands, production authorizations, and parts is completely coupled.

Special case of the KCS: CONWIP control system

It is worth noting that the *CONWIP Control System* [20] is a special case of a single-stage KCS. The CONWIP Control System was initially presented in the context of a production line. A production line is a manufacturing system consisting of several machines in series with buffers between them. The idea of CONWIP control is that as soon as a finished product leaves a production line to be shipped to a customer, a new part enters the production line to begin its processing. This implies that the total WIP in the system, including the finished goods inventory, is a constant; hence the name CONWIP (CONstant WIP).

In its introduction CONWIP was presented as an alternative to Kanban control rather than as a special case of it (see [20]). This is because the two systems were compared at different levels of stage decomposition. Namely, the Kanban Control System was seen as a production control system where control was applied at each

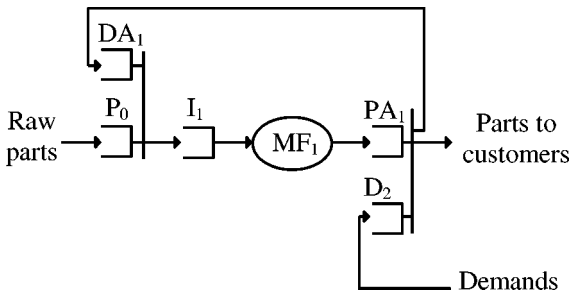

Figure 6. A CONWIP Control System.

and every machine of the production line, whereas the CONWIP Control System was seen as a production control system where control was applied at the entry of the production line.

We define the CONWIP to be a pull control mechanism applied to any manufacturing system whereby as soon as a finished product leaves the manufacturing system to be shipped to a customer, a new part enters the manufacturing system to begin its processing. Under this definition, a CONWIP Control System is equivalent to a single-stage KCS that uses Kanban control to release parts into and out of that system (see figure 6).

3.2. Two-parameter-per-stage pull control systems

Next, we present the Generalized Kanban and the Extended Kanban Control Systems. Each system depends on two parameters per stage and includes both the Base Stock and the Kanban Control Systems as special cases.

3.2.1. Generalized Kanban Control System

The KCS was generalized into a system called the *Generalized Kanban Control System* (GKCS) [5,22]. The GKCS was proposed as a general approach to pull production control incorporating the Kanban and the Base Stock systems. The GKCS depends on two parameters for each Stage i, the number of kanbans, K_i, and the base stock of parts in inventory, S_i. A broadened version of the GKCS that includes two more parameters per stage, one concerning the lot-sizing of parts and the other concerning time-delays, called the PAC system, was developed in [6,7]. Batching of parts and time-delays, which are related to manufacturing lead times, fall outside the scope of this paper. In this section, therefore, we focus on the basic two-parameter-per-stage GKCS.

In the original presentation of the GKCS [5] there was no restriction on parameters K_i and S_i other than that $K_i > 0$ and $S_i \geqslant 0$, for all i; however, a distinction between two cases was made. In the first case, $K_i \geqslant S_i$, for all i, and in the second case, $K_i < S_i$, for all i. A GKCS with $K_i \geqslant S_i$ was called a *Backordered Kanban System*, and a GKCS with $K_i < S_i$ was called a *Reserve Stock Kanban System*. Actually, as is discussed in appendix C, the GKCS with $K_i < S_i$,

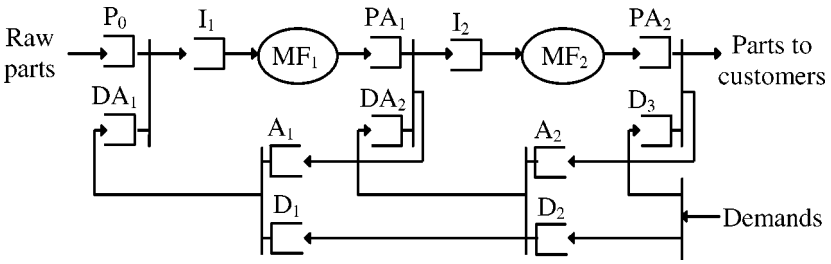

Figure 7. A GKCS with two stages in series.

for all i, is equivalent to a KCS with IM-WIP-control and is presented separately in section 5.1. In this section, therefore, we only concentrate on the GKCS with $K_i \geqslant S_i$, for all i.

The original queueing network model of the GKCS, that allows for both cases, $K_i \geqslant S_i$ and $K_i < S_i$, is shown in figure 20 in appendix C. The behavior of the system in figure 20 for the case $K_i \geqslant S_i$, for all i, however, is exactly the same as that of the system shown in figure 7, as is discussed in appendix C. We can therefore use either system to illustrate the GKCS for the case $K_i \geqslant S_i$, for all i. In this paper, we prefer to use the system in figure 7, over the system in figure 20, because the first system is consistent with our definition of a pull control mechanism in section 2. According to this definition, a production authorization associated with a stage is detached from a part after that part leaves the output buffer of that stage, whereas the second system is not consistent with that definition.

Queue PA_i, in figure 7, represents the output buffer of Stage i, $i = 1, 2$ and contains pairs of Stage-i finished parts and production authorizations, $i = 1, 2$. Queue DA_i contains pairs of production authorizations and demands for the production of new Stage-i finished parts, $i = 1, 2$. Queue A_i contains free Stage-i kanbans, $i = 1, 2$. Queue D_i contains demands for the production of new Stage-i finished parts, $i = 1, 2$. Queue P_0 represents the raw-parts buffer, and Queue D_3 contains customer demands. Queue I_i represents the input buffer of Stage i, $i = 1, 2$.

When the system is in its initial state, that is, before any demands have arrived to the system, P_0 contains an initial number of raw parts, PA_i contains S_i Stage-i finished parts, each part having a Stage-i kanban attached to it, A_i contains $K_i - S_i$ Stage-i kanbans, $i = 1, 2$, and all other queues in the system are empty. S_i and K_i are the only control parameters of Stage i. S_i is referred to the base stock of Stage i. It is assumed that $K_i \geqslant S_i$, for all i.

We will describe the operation of the GKCS by focusing on how it differs from the operation of the KCS. In both the KCS and the GKCS a demand for the production of a new Stage-i part is carried upstream to DA_i by a Stage-i kanban. The difference between the two systems is that in the KCS, initially, all Stage-i kanbans are attached to an equal number of finished parts in PA_i (see figure 5), that is, there are no free kanbans. When a Stage-i finished part leaves Stage i, the Stage-i kanban that was attached to it is freed and immediately carries a demand upstream to DA_i; hence,

the complete coupling between the transfer of demands, kanbans, and parts, that was mentioned earlier.

In the GKCS, on the other hand, initially, there are S_i kanbans attached to an equal number of finished parts in PA_i, but there are also $K_i - S_i$ free kanbans in A_i (see figure 7). These extra kanbans allow for the partial decoupling of the transfer of parts downstream of Stage i and the transfer of demands upstream to DA_i. In the special case where $S_i = K_i$, for all i, the GKCS is equivalent to the KCS with parameters K_i.

The use of two parameters for every Stage i, S_i and K_i, renders the GKCS a significant improvement over the KCS as it loosens the coupling between the transfer of production authorizations and demands and the release of parts that is present in the KCS. This means that the GKCS can respond faster than the KCS to customer demands as the number of extra kanbans, $K_i - S_i$, increases. In the limiting case when $K_i = \infty$, for all i, the GKCS is equivalent to the BSCS with the same base stock parameters S_i as those of the GKCS.

3.2.2. Extended Kanban Control System

The *Extended Kanban Control System* (EKCS) [10,11] was proposed as a general approach to pull production control combining the Base Stock and Kanban Control Systems. The EKCS, like the GKCS, depends on two parameters per stage, the number of kanbans, K_i, and the base stock of parts in inventory, S_i.

Figure 8 shows the queueing network model of an EKCS with two stages in series. Queue PA_i represents the output buffer of Stage i and contains pairs of Stage-i finished parts and production authorizations, $i = 1, 2$. Queue A_i contains free Stage-i kanbans, $i = 1, 2$. Queue D_i contains demands for the production of new Stage-i finished parts, $i = 1, 2$. Queue P_0 represents the raw-parts buffer, and Queue D_3 contains customer demands. Queue I_i represents the input buffer of Stage i, $i = 1, 2$.

When the system is in its initial state, that is, before any demands have arrived to the system, P_0 contains an initial number of raw parts, PA_i contains S_i Stage-i finished parts, each part having a Stage-i kanban attached to it, A_i contains $K_i - S_i$ Stage-i kanbans, $i = 1, 2$, all other queues in the system are empty. S_i and K_i are the only

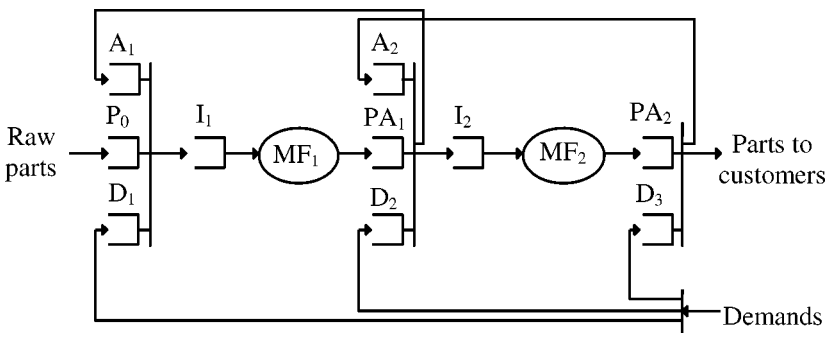

Figure 8. An EKCS with two stages in series.

control parameters of Stage i. S_i is referred to the base stock of Stage i. It is assumed that $K_i \geqslant S_i$, for all i.

The dynamics of the EKCS are a combination of the dynamics of the BSCS and the KCS. In the EKCS, when a customer demand arrives to the system, it joins Queue D_3, thereby requesting the release of a finished part from PA_2 to the customer. The customer demand instantly also generates a demand in D_i, $i = 1, 2$, as is the case in the BSCS. Unlike the case of the BSCS, however, a part is not immediately authorized to be released from PA_{i-1} to I_i, $i = 1, 2$, unless there is a free kanban in A_i, as is the case in the KCS. Initially, there are $K_i - S_i$ free kanbans in A_i which may authorize an equal number of new parts to be released into Stage i; however, in order to authorize the release of any part above this number, a finished Stage-i part must leave Stage i.

In the limiting case when $K_i = \infty$, for all i, the EKCS is equivalent to the BSCS with the same base stock parameters S_i as those of the EKCS. This is because when there is an infinite number of kanbans in each stage, Queues A_i are redundant and may be removed from the system in figure 8. In the special case when $S_i = K_i$, for all i, the EKCS is equivalent to the KCS with parameters K_i. This is because when the number of kanbans is equal to the base stock level in every stage, the queues containing demands for the production of new parts into each stage are redundant and may be removed from the system in figure 8 (see [11]).

The philosophy of the EKCS is that when a customer demand arrives to the system, it is immediately broadcast to every stage in the system, as is the case in the BSCS. Unlike what happens in the BSCS, however, a part is actually authorized to be released from one stage to the downstream stage only if one of a finite number of production authorizations or kanbans associated with that stage is available, as is the case in the KCS.

The EKCS, like the GKCS, depends on two parameters per stage, S_i and K_i, and includes the BSCS and the KCS as special cases. Both the EKCS and the GKCS operate under the restriction that $K_i \geqslant S_i$ since, as was mentioned earlier, the GKCS with $K_i < S_i$ is a special case of a KCS with IM-WIP-control, which can also be viewed as a special case of an EKCS with IM-WIP-control (see section 5.2). The EKCS, however, is much simpler than the GKCS. To better understand the difference between the two systems, we have included a short discussion in appendix D based on an alternative queueing network model of the EKCS shown in figure 21.

Another advantage of the EKCS over the GKCS, besides simplicity, is that in the ECKS the role of parameters S_i and K_i is clearly distinguishable, whereas in the GKCS it is not. A consequence of this is that the production capacity of the EKCS, that is, the maximum average demand rate it can meet, does not depend on parameters S_i but only depends on parameters K_i. This is because the so-called saturated EKCS (i.e., the EKCS with an infinite number of demands), whose throughput is the production capacity of the EKCS, is equivalent to the saturated KCS, and the throughput of the latter only depends on parameters K_i (see [11]). The production capacity of the GKCS, on the other hand, depends on both parameters S_i and K_i. This opens the possibility

to design the EKCS in two separate steps: First, design K_i for every Stage i so as to achieve a desired production capacity level, and then design S_i for every Stage i to reach a satisfactory customer service level.

4. Stage-WIP-control systems

In section 3 we presented four basic pull control systems, each representing a different approach to stage coordination, that is, the coordination of the release of parts into every stage with the arrival of customer demands for final products at the end of the system. The four different approaches to stage coordination were depicted as different ways of interconnecting the synchronization stations between the stages and the point where customer demands arrive, on the respective queueing network models.

In this section we argue that on top of any of these four pull production systems, it is always possible to impose additional mechanisms to control the WIP in each stage. These additional mechanisms will be part of local production control within a stage as opposed to global stage coordination.

Figures 9–11 show three stage-WIP-control mechanisms modeled as closed queueing networks. In each mechanism, the closed queueing network includes an area of the stage in which WIP is controlled and a queue called C_i. Every Stage i has associated with it W_i *WIP-control authorizations* which correspond to the maximum

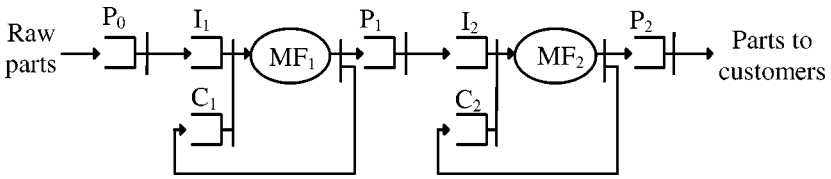

Figure 9. A two-stage manufacturing system with M-WIP-control in every stage.

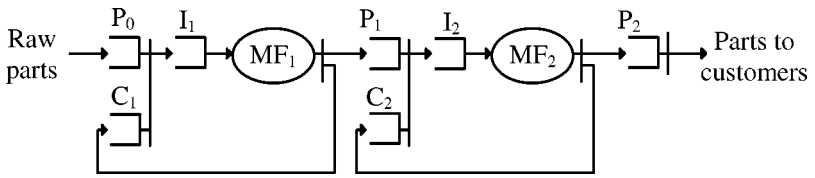

Figure 10. A two-stage manufacturing system with IM-WIP-control in every stage.

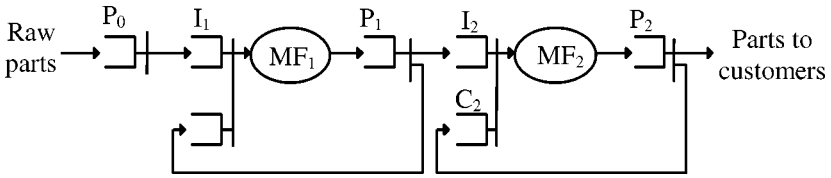

Figure 11. A two-stage manufacturing system with MO-WIP-control in every stage.

number of parts allowed in the WIP-controlled area of that stage. Every part in the WIP-controlled area of Stage i has a WIP-control authorization attached to it. A WIP-control authorization is freed whenever a part leaves the WIP-controlled area. Free Stage-i WIP-control cards are stored in Queue C_i indicating an equal number of empty spaces in the WIP-controlled area of Stage i. Note that in figures 9–11 no demands or synchronization stations between stages are show. This is because the focus is on stage-WIP-control and not on the coordination between stages.

In the three cases shown in figures 9–11, the areas in each stage where WIP is controlled are: the manufacturing facility, the input buffer and the manufacturing facility, and the manufacturing facility and the output buffer, respectively. We call these three cases, *M-WIP-control*, *IM-WIP-control*, and *MO-WIP-control*, respectively.

The case of WIP-control in the entire stage, that is, the input buffer, the manufacturing facility and the output buffer, is not included, since this case would be classified as stage coordination – in fact Kanban coordination (see figure 5) – rather than stage-WIP-control. This is because in this case the release of parts into one stage directly affects the release of parts into the upstream stage.

5. Pull control systems with stage-WIP-control

In this section we argue that it is always possible to superimpose one or more of the stage-WIP-control mechanisms presented in section 4 on one of the four basic pull control systems presented in section 3 and create a new control system. These additional mechanisms will be part of local production control within a stage as opposed to global stage coordination provided by the basic pull control systems. To illustrate this, we present and discuss two examples.

5.1. KCS with IM-WIP-control

Figure 12 shows the queueing network model of a KCS with IM-WIP-control with two stages in series. The system in figure 12 is obtained by superimposing the IM-WIP-control system in figure 10 on top of the KCS in figure 5.

The initial conditions of the KCS with IM-WIP-control are exactly the same as those of the KCS, as far as the KCS part of the system is concerned. In addition, C_i initially contains W_i WIP-control authorizations, $i = 1, 2$. K_i and W_i are the only control parameters of Stage i. It is assumed that $W_i < K_i$, for all i, otherwise the IM-WIP-control mechanism is redundant.

The conditions to release a part from the output buffer of Stage $i-1$ into the input buffer of Stage i are the sum of the respective conditions in the KCS and in the IM-WIP-control mechanism. These are that there must be a part plus kanban in PA_{i-1}, a demand plus kanban in DA_i, and a WIP-control authorization in C_i.

It is worth noting that the KCS with IM-WIP-control is equivalent to the so-called *Local Control System* (LCS) [7]. The LCS is a pull control system in which the

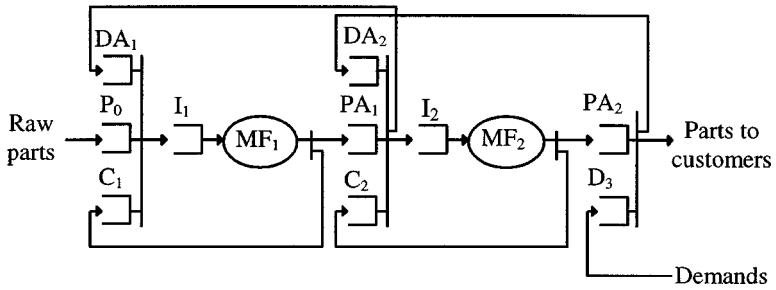

Figure 12. A KCS with IM-WIP-control with two stages in series.

conditions to release a part from the output buffer of a stage into the input buffer of the downstream stage are the following:

(1) such a part exists,

(2) the input buffer and the manufacturing facility of the following stage are not full, and

(3) the input buffer, the manufacturing, and the output buffer of the following stage, combined, are not full.

This system is called *local* control because the decision to release a part into a stage uses no information from downstream stages apart from the availability of space in the stage.

It is also worth noting that the KCS with IM-WIP-control with parameters K_i and W_i, with $W_i < K_i$, for all i, is equivalent to a GKCS whose parameters S_i and K_i, GKCS are equal to parameters K_i and W_i of the KCS with IM-WIP-control, respectively, and satisfy $K_i < S_i$, for all i. To see why this is the case, it suffices to say that it is shown in [7] that the GKCS with $K_i < S_i$, for all i, is equivalent to the LCS, which, as was already mentioned, is equivalent to the KCS with IM-WIP-control in which $W_i < K_i$, for all i.

5.2. EKCS with IM-WIP-control

Figure 13 shows the queueing network model of an EKCS with IM-WIP-control. The system in figure 13 is obtained by superimposing the IM-WIP-control system in figure 10 on top of the EKCS in figure 8.

The initial conditions of the EKCS with IM-WIP-control in figure 13 are exactly the same as those of the EKCS, as far as the EKCS part of system is concerned. In addition, C_i initially contains W_i WIP-control authorizations, $i = 1, 2$. S_i, K_i, and W_i are the only control parameters of Stage i. It is assumed that $K_i \geqslant S_i$ and that $K_i > W_i$, for all i.

The EKCS with IM-WIP-control system is a general system that includes the EKCS, the KCS with or without IM-WIP-control, and the BSCS with or without IM-WIP-control as special cases, with the appropriate choice of parameters S_i, K_i, and W_i. For example, the EKCS with IM-WIP-control with $W_i \geqslant K_i$, for all i, is clearly equivalent to the EKCS, because, when $W_i \geqslant K_i$, for all i, the WIP-

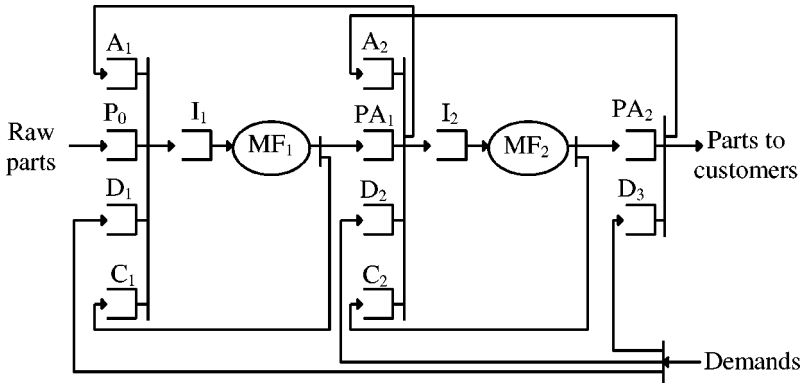

Figure 13. An EKCS with IM-WIP-control with two stages in series.

control mechanisms become redundant. Also, the EKCS with IM-WIP-control with parameters $S_i = K_i > W_i$, for all i, is equivalent to the KCS with IM-WIP-control with the same parameters K_i and W_i as those of the EKCS with IM-WIP-control. The EKCS with IM-WIP-control with $S_i = K_i > W_i$, for all i, is also equivalent to a GKCS whose parameters S_i and K_i are such that $K_i < S_i$, for all i, and are equal to the parameters K_i and W_i of the EKCS with IM-WIP-control, respectively.

6. Single-machine-per-stage production control systems

In this section we focus on single-machine-per-stage pull production control systems, that is, production control mechanisms where control is exercised on each and every machine of the physical manufacturing system. As in the previous sections, we restrict our attention to serial systems, which in this case means several machines in series having buffers in between them. Such systems are referred to as production lines, transfer lines, or manufacturing flow lines, and have attracted a lot of attention in the literature. For a recent review on production lines see [9].

We argue that we can apply any of the pull control mechanisms presented in the previous sections to coordinate the loading of parts on each machine of a single-machine-per-stage manufacturing system. To illustrate this, we present four examples. All of these examples, except the one in section 6.3, are special cases of systems that have already been presented in previous sections, and have the same initial conditions and behavior as these systems. For this reason we will not repeat their initial conditions and the detailed description of their behavior.

The first two examples, in particular, the single-machine-per-stage KCS, presented in section 6.1, and the single-machine-per-stage KCS with IM-WIP-control, presented in section 6.2, represent two variations of what has often been called Kanban control in the literature. We hope that this exposition will help clarify some of the confusion around the issue of what Kanban control is.

Another issue that has somewhat added to the confusion around Kanban control is that some authors have used blocking mechanisms applied on production lines with

finite buffers to describe Kanban control (e.g., see [2]). To help clarify matters, we point out the equivalence between several blocking mechanisms and single-machine-per-stage production systems.

6.1. Single-machine-per-stage KCS

The queueing network model of a single-machine-per-stage KCS is a special case of the KCS in figure 5, in which the manufacturing facility at each Stage i, MF_i, represents a single machine, M_i.

It is worth noting that a production line operated under the so-called *Minimal Blocking* mechanism [18] is equivalent to the single-machine-per-stage KCS. Minimal blocking is a blocking mechanism applied on a production line of machines, where each machine M_i has an Input/Output buffer B_i containing parts that are waiting to be processed by the machine and parts that have been processed by the machine but are unable to move to the downstream buffer because there is no space in it. The capacity of machine M_i is one, and the capacity of buffer B_i is K_i including the space in the machine. More specifically, a part is able to enter the buffer of a machine if there is space in it. Having entered, the part waits in the buffer, and when its turn comes, it is loaded on the machine to receive processing. At the instant of completion of a part on the machine, the part is released from the machine and either enters the buffer of the downstream machine, if there is space in it, or is stored in the same buffer from where it was loaded. In any case, the machine is free to start a new service, provided that there is a part requiring one. A machine is blocked when its buffer is full of parts that have been processed by it but are unable to move to the downstream buffer because there is no space in it.

Seen in the context of Minimal Blocking, the queues in the queueing network model in figure 5 have the following meaning. Queue PA_i contains parts that have received processing in machine M_i and are stored in buffer B_i because they are unable to move to the downstream buffer B_{i+1} (or to the customer, in the case of $i = 2$), $i = 1, 2$. Queue I_i contains parts that are stored in buffer B_i and are waiting to be processed by machine M_i, $i = 1, 2$. Queue DA_i contains production authorizations that represent the available spaces in buffer B_i, $i = 1, 2$.

6.2. Single-machine-per-stage KCS with IM-WIP-control

The queueing network model of a single-machine-per-stage KCS with IM-WIP-control is a special case of the KCS with IM-WIP-control system in figure 12, in which the manufacturing facility at each Stage i, MF_i, represents a single machine, M_i, with unit capacity.

The IM-WIP-control loops in figure 12 are closed queueing networks, each having W_i customers. Two extreme cases are the cases (1) $W_i \geqslant K_i$, for all i, and (2) $W_i = 1$, for all i. The first case is equivalent to the single-machine-per-stage KCS shown in figure 5. This is because when $W_i \geqslant K_i$, for all i, the IM-WIP loops in figure 12 become redundant and may be dropped. The second case is equivalent to

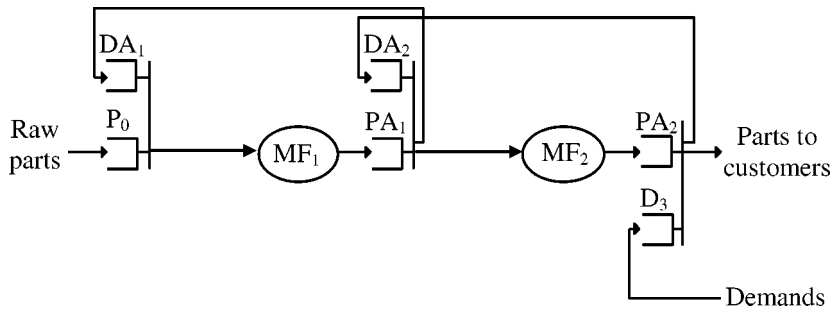

Figure 14. A single-machine-per-stage KCS with unit IM-WIP-control with two stages in series.

the system shown in figure 14, which we call single-stage KCS with *unit* IM-WIP-control. This is because when $W_i = 1$, for all i, the capacity of input buffer I_i and machine M_i, combined, is one part, which means that M_i has a capacity of one and I_i has a capacity of zero, i.e., that there is no input buffer.

The difference between the single-stage KCS in figure 5 and the single-stage KCS with unit IM-WIP-control in figure 14 is that, in the first system, each machine has an input buffer, whereas in the second system there are no input buffers. Consequently, in the first system, a Stage-i kanban is released from a part when that part is consumed by the downstream input buffer I_{i+1} but before it is loaded onto the downstream machine M_{i+1}. In the second system, on the other hand, a Stage-i kanban is released from a part only when that part is consumed by (i.e., loaded onto) machine M_{i+1}.

We explicitly single out these two cases because either case has at times been referred to as Kanban control in the literature (e.g., see [2]). We want to emphasize that we view Kanban control as being the control system shown in figure 5 and that the systems shown in figures 12 and 14 are an extension and a variation of that extension, respectively, of the system shown in figure 5.

It is worth noting that a production line operated under the so-called *blocking before service with place nonoccupied* (BBS-PNO) mechanism (e.g., see [9]) is equivalent to the single-machine-per-stage KCS with unit IM-WIP-control. BBS-PNO is a blocking mechanism applied on a production line of machines, where each machine M_i has an output buffer B_i, that is, a buffer containing parts that have been processed by the machine, and no input buffer. The capacity of machine M_i is one, and the capacity of buffer B_i is K_i. In BBS-PNO, a machine can start processing a part only if there is a space available in the downstream buffer. Otherwise it has to wait until a space becomes available. A machine is blocked when the downstream buffer is full. While the machine is blocked, the position (space) on the machine may not be occupied.

Seen in the BBS-PNO context, the queues in the queueing network model in figure 14 have the following meaning. Queue PA_i represents the output buffer B_i of machine M_i, $i = 1, 2$, and Queue DA_i contains production authorizations that represent the available spaces in buffer B_i, $i = 1, 2$.

Another blocking mechanism that is often associated with kanban systems is blocking after service (BAS) (e.g., see [9]). BAS is a blocking mechanism applied to a production line of machines, where each machine M_i has an output buffer B_i. BAS occurs if at the instant of completion of a part on a machine, the downstream buffer is full. In this case, the part stays on the machine until a space is available in the downstream buffer. During this time the machine is prevented from working and is blocked. When a space becomes available in the downstream buffer, the part is immediately transferred and the machine can start processing another part, if any. In [12], it is shown that BAS with buffer capacity $K_i - 1$ is equivalent to BBS-PNO with buffer capacity K_i, for all i; therefore, a production line operated under BAS with buffer capacities $K_i - 1$, for all i, is equivalent to the single-machine-per-stage KCS with unit IM-WIP-control and kanban parameters K_i, for all i.

6.3. *Single-machine-per-stage KCS with IM-WIP and MO-WIP-control*

Figure 15 shows the queueing network model of a single-machine-per-stage KCS with IM-WIP and OM-WIP-control. Figure 15 is obtained by superimposing the IM-WIP-control system in figure 10 and the MO-WIP-control system in figure 11 on top of the KCS in figure 5, and replacing the manufacturing facility at each Stage i, MF_i, by a single machine, M_i. Note that Queues C_i in the OM-WIP-control system in figure 10 have been relabeled as G_i in the new hybrid system in figure 15.

In the single-machine-per-stage KCS with IM-WIP and OM-WIP-control, each Stage i has three parameters associated with it: the number of kanbans, K_i, the number of IM-WIP-control authorizations, W_i, and the number of OM-WIP-control authorizations, S_i. Parameters W_i, S_i, and K_i are respectively the upper limits on the IM-WIP, MO-WIP, and total WIP in Stage i.

The initial conditions of the single-machine-per-stage KCS with IM-WIP and MO-WIP-control are as follows. P_0 contains an initial number of raw parts, PA_i contains S_i Stage-i finished parts, each part having a Stage-i kanban and an OM-WIP-control authorization attached to it, $i = 1, 2$. I_i contains $K_i - S_i$ parts, each part having a Stage-i kanban and an IM-WIP-control authorization, $i = 1, 2$. C_i contains

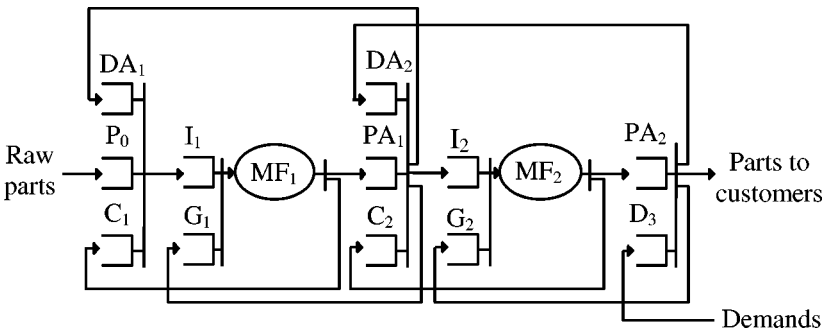

Figure 15. A single-machine-per-stage KCS with IM-WIP and MO-WIP-control with two stages in series.

$W_i + S_i - K_i$ IM-WIP-control authorizations, $i = 1, 2$. It is assumed that $K_i \geqslant W_i$, $K_i \geqslant S_i$, and $W_i + S_i \geqslant K_i$, otherwise there will be some WIP-control limits that will never be reached and the respective WIP-control mechanism will be redundant. All the other queues in the network are empty.

It is worth noting that a production line operated under the so-called *general blocking* [8] is equivalent to the single-machine-per-stage KCS with IM-WIP and OM-WIP-control. General blocking is a blocking mechanism applied to a production line of machines, where each machine M_i has an input buffer containing parts that are waiting to be processed by the machine and an output buffer containing parts that have been processed by the machine but are unable to move to the downstream buffer either because there is no space in it or because the total number of parts in the downstream stage has reached its upper limit. In general blocking, a machine can start processing a part only if there is a space available in the downstream buffer. Otherwise it has to wait until a space becomes available.

Seen in the general blocking context, the queues in the queueing network model in figure 15 have the following meaning. Queue PA_i represents the output buffer of machine M_i, $i = 1, 2$. Queue I_i represents the input buffer of machine M_i, $i = 1, 2$. Queue DA_i contains production authorizations that represent the available spaces in Stage i, $i = 1, 2$. Queue C_i contains IM-WIP-control authorizations that represent the available spaces in the input buffer and machine at Stage i, $i = 1, 2$. Queue G_i contains OM-WIP-control authorizations that represent the available spaces on the machine and in the output buffer at Stage i, $i = 1, 2$. It should be noted that in the Generalized Kanban terminology, parameters K_i, W_i, and S_i, are called k_i, a_i, and b_i, respectively.

6.4. Single-machine-per-stage EKCS with IM-WIP-control

The queueing network model of a single-machine-per-stage EKCS with IM-WIP-control is a special case of the EKCS with IM-WIP-control shown in figure 13, in which the manufacturing facility at each Stage i, MF_i, represents a single machine, M_i. Alternatively, this control mechanism can be obtained by superimposing the single-machine-per-stage KCS with IM-WIP-control in figure 12 on the BSCS in figure 4.

It is worth noting that a production control scheme recently proposed in [14], that superimposes a HPCS on top of a production line operated under BBS-PNO, is equivalent to a single-machine-per-stage EKCS with unit IM-WIP-control, that is, a single-machine-per-stage EKCS with IM-WIP-control with $W_i = 1$, for all i. This system is shown in figure 16. To see this equivalence, recall that (1) the single-machine-per-stage EKCS with unit IM-WIP-control is equivalent to a single-machine-per-stage KCS with unit IM-WIP-control, shown in figure 14, superimposed on a BSCS, shown in figure 4, and (2) the single-machine-per-stage KCS with unit IM-WIP-control is equivalent to a production line operated under BBS-PNO, and the BSCS is equivalent to the HPCS.

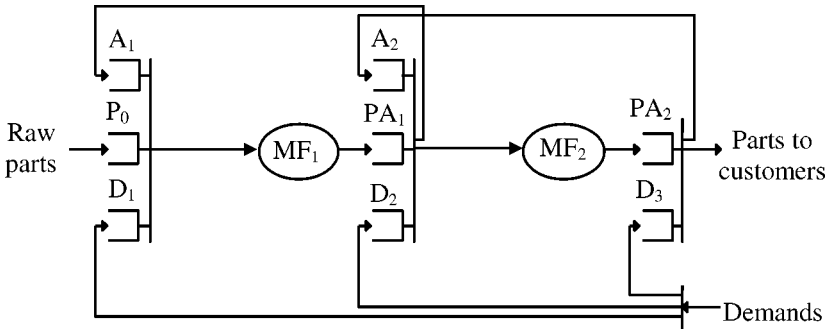

Figure 16. A single-machine-per-stage EKCS with unit IM-WIP-control with two stages in series.

7. Nesting of pull production control systems

In this final section we argue that it is possible to nest several pull control systems to create new control systems. By nesting we mean specifying a hierarchy of stages where higher-level stages in the hierarchy are formed by aggregating several lower-level stages together, and where a different pull production control mechanism is used at each stage level. This could be done for at least two reasons. Firstly, it may be natural to want to coordinate higher-level stages in one way, e.g., using a BSCS to emphasize quick response to demands, and the lower-level stages in another way, e.g., using a KCS to emphasize WIP-control. Secondly, it may be easier to design such mechanisms by decomposing the decision on the control parameters according to the different hierarchy levels. To illustrate what we mean by nesting, we present two examples.

7.1. KCS containing nested single-machine-per-stage KCSs with unit IM-WIP-control

Figure 17 shows the queueing network model of a manufacturing system with four machines in series, with two levels of stage coordination. The higher level has two stages. The first high-level stage contains machines M_1 and M_2, and the second high-level stage contains machines M_3 and M_4. Kanban control is used to release parts into each high-level stage.

Each high-level stage is further divided into two low-level stages. Each low-level stage contains a single machine. Kanban control with unit IM-WIP-control is used to release parts into each low-level stage. The parameters of the system are the number of kanbans in each of the four low-level stages and the number of kanbans in each of the two high-level stages.

It is worth noting that a production line operated under the so-called *CONWIP/Kanban Hybrid Control System* [3,4] is a special case of a KCS containing nested single-machine-per-stage KCSs with unit IM-WIP-control, in which the high-level KCS has only one stage, that is, it is a CONWIP system. In other words, the CONWIP/Kanban Hybrid Control System is a system where CONWIP control is used to

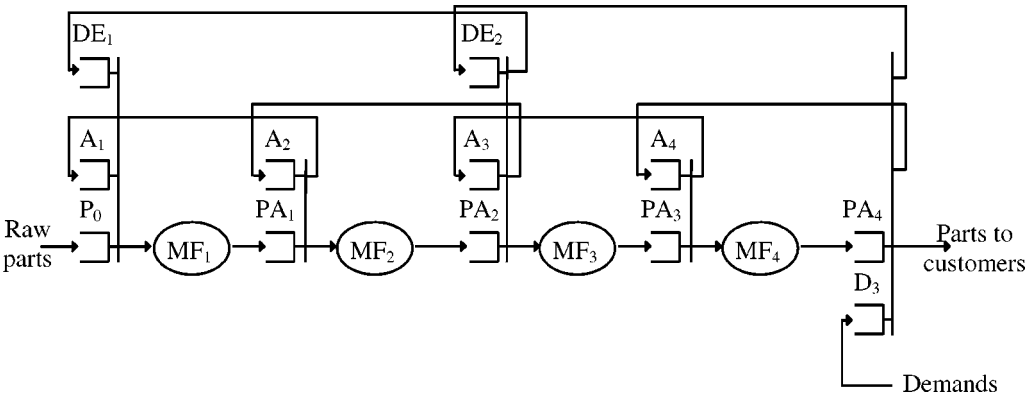

Figure 17. A KCS with two stages in series, each stage containing a nested single-machine-per-stage KCS with unit IM-WIP-control with two stages in series.

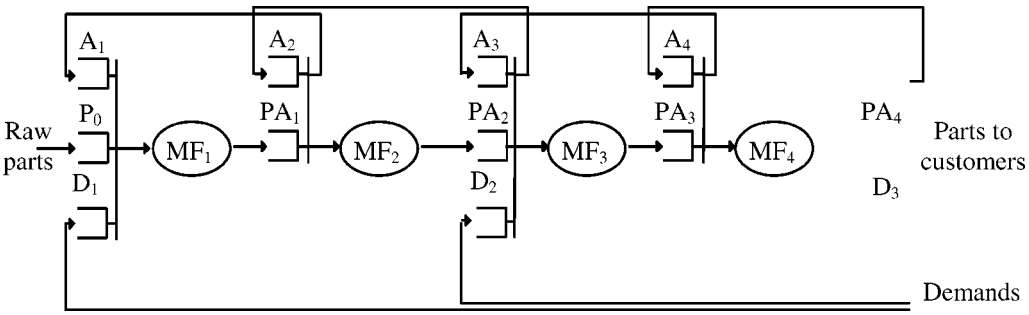

Figure 18. A BSCS with two stages in series, each stage containing a nested single-machine-per-stage KCS with IM-WIP-control with two stages in series.

release parts into the system, and Kanban control with unit IM-WIP-control is used to release parts into each single-machine stage.

7.2. BSCS containing nested single-machine-per-stage KCSs with unit IM-WIP-control

Figure 18 shows the queueing network model of a manufacturing system with four machines in series, with two levels of stage coordination. The higher level has two stages. The first high-level stage contains machines M_1 and M_2, and the second high-level stage contains machines M_3 and M_4. Base Stock control is used to release parts into each high-level stage.

Each high-level stage is further divided into two low-level stages. Each low-level stage contains a single machine. Kanban control with unit IM-WIP-control is used to release parts into each low-level stage. The parameters of the system are the number of kanbans in each of the four low-level stages and the base stock levels in each of the two high-level stages.

It is worth noting that a production line operated under the so-called *Base Stock/Kanban Hybrid Control System* [3,4] is a special case of a BSCS containing nested single-machine-per-stage KCSs with unit IM-WIP-control, in which the high-level BSCS has only one stage. In other words, the Base Stock/Kanban Hybrid Control System is a system where Base Stock control is used to release parts into the system, and Kanban control with unit IM-WIP-control is used to release parts into each single-machine stage.

8. Conclusions

We presented a unified framework for pull production control mechanisms in multi-stage manufacturing systems. In this framework, a pull production control mechanism is a mechanism that coordinates the release of parts into each stage of a manufacturing system that has been partitioned into several stages, with the arrival of customer demands for final products. First, four basic stage coordination systems were presented. These were the BSCS, KCS, GKCS, and EKCS. The BSCS and KCS each depend on one parameter per stage, whereas the GKCS and EKCS each depend on two parameters per stage. Then, we argued that on top of each of these stage coordination mechanisms it is possible to superimpose a local mechanism to control the WIP within each stage. Several cases of basic stage coordination mechanisms with local WIP-control were presented, and several production control systems that have appeared in the literature were shown to be equivalent to some of these cases.

What we tried to do here is propose a unified way for defining such mechanisms. Considerable work needs to be done to be able to evaluate the performance of and optimize such mechanisms. Some numerical results on the performance of the Generalized Kanban system have been reported in [16,21]. Certainly, much more numerical work needs to be done.

Acknowledgement

The authors are grateful to the anonymous referees for helpful suggestions that improved the presentation of the paper.

Appendix A. Integral control system

A pull production control system that, like the Base Stock Control System (BSCS) and the Kanban Control System (KCS), depends on one parameter per stage is the so-called *Integral Control System* (ICS) [7]. In fact, the ICS borrows elements from both the BSCS and the KCS. More precisely, in the ICS, when a customer demand arrives at the last stage, it is transmitted upstream of that stage only when a finished product is consumed by a customer, as is the case in the KCS; however, when the demand is

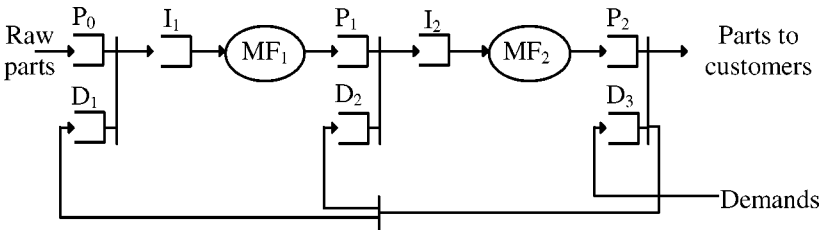

Figure 19. Model of an ICS with two stages in series.

transmitted upstream of the last stage, it is transmitted to all stages at once, as is the case in the BSCS.

Figure 19 shows the queueing network model of an *Integral Control System* (ICS) [7] with two stages in series. Queue P_i represents the output buffer of Stage i, $i = 1, 2$. Queue D_i contains demands for the production of new Stage-i finished parts, $i = 1, 2$. Queue P_0 represents the raw-parts buffer, and Queue D_3 contains customer demands. Queue I_i represents the input buffer of Stage i, $i = 1, 2$.

When the system is in its initial state, that is, before any demands have arrived to the system, P_0 contains an initial number of raw parts, P_i contains S_i Stage-i finished parts, $i = 1, 2$, and all other queues in the system are empty. S_i is the only control parameter of Stage i.

It is worth noting that the ICS can also be seen as a special case of a GKCS with $K_i \geqslant K_{i+1} + S_i$, for all Stages i except the last stage, and $K_N = S_N$, for the last Stage N (see [7]).

Appendix B. Proof of theorem 1

Let B_i denote the number of parts in Stage i, i.e.,

$$B_i = M(I_i) + M(MP_i) + M(P_i), \quad i = 1, \ldots, N,$$

where $M(Q)$ denotes the number of customers in any queue or sub-network of queues, Q, in the system. Actually, the number of customers in any queue or sub-network of queues also depends on time, but for notational simplicity we omit this dependence here, because all the relations involving this number, that we develop, hold at all times.

Also, suppose that unsatisfied customer demands are stored in a queue called D_4, whatever the pull control system is.

Let X_i denote the inventory/backlog position of Stage i, i.e., the difference between the cumulative number of parts that have been released for production into Stage i and the cumulative number of customer demands that have arrived to the manufacturing system, whatever the pull control mechanism is. The cumulative number of parts that have been released into Stage i is equal to the number of parts that are in the system downstream of P_{i-1} plus the cumulative number of parts that have been released to the customer. Similarly, the cumulative number of customer demands that have arrived to the system is equal to the number of unsatisfied customer demands

plus the cumulative number of customer demands that have been satisfied, which is equal to the cumulative number of parts that have been released to the customer. X_i is therefore equal to the number of parts that are in the system downstream of P_{i-1} minus the number of unsatisfied customer demands. Given the definitions of B_i and D_4, X_i can be written as

$$X_i = \sum_{k=i}^{N} B_k - M(D_{N+1}), \quad i = 1, \ldots, N, \tag{1}$$

whatever the pull control system is.

In the HPCS the necessary and sufficient condition for releasing a part into Stage i, provided such a part exists upstream of Stage i, is

$$X_i < Z_i, \quad i = 1, \ldots, N. \tag{2}$$

To prove the theorem we need to show that condition (2) also holds for a BSCS with parameters $S_i = Z_i - Z_{i+1}$, $i = 1, \ldots, N$. For a BSCS with N stages in series, the following holds:

$$M(D_i) + \sum_{k=i}^{N} B_k - M(D_{N+1}) = \sum_{k=i}^{N} S_k, \quad i = 1, \ldots, N. \tag{3}$$

By combining (1) and (3) we can write the following expression for X_i for the BSCS:

$$X_i = \sum_{k=i}^{N} S_k - M(D_i), \quad i = 1, \ldots, N. \tag{4}$$

Suppose the BSCS parameters are given by

$$S_i = Z_i - Z_{i+1}, \quad i = 1, \ldots, N, \tag{5}$$

where the Z_i, $i = 1, \ldots, N$, are the HPCS parameters and $Z_{N+1} = 0$ by convention. Substituting (5) into (4) yields

$$X_i = Z_i - M(D_i), \quad i = 1, \ldots, N. \tag{6}$$

In the BSCS the necessary and sufficient condition for releasing a part into Stage i, provided such a part exists upstream of Stage i, is

$$M(D_i) > 0, \quad i = 1, \ldots, N,$$

which, after substituting $M(D_i)$ from (6), becomes

$$X_i < Z_i, \quad i = 1, \ldots, N.$$

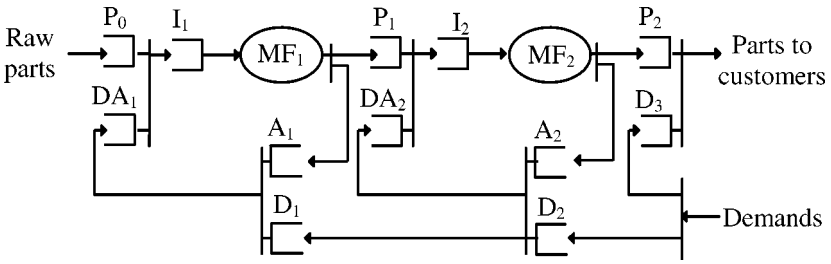

Figure 20. Original model of a GKCS with two stages in series.

Appendix C. Original queueing network model of the GKCS

The original queueing network model of the GKCS [5] is shown in figure 20. The initial conditions in that network are that P_0 contains an initial number of raw parts, P_i contains S_i parts, A_i contains K_i production authorizations, $i = 1, 2$, and all other queues in the network are empty.

It can be shown that when $K_i \geqslant S_i$, for all i, the behavior of the system shown in figure 20 is exactly the same as that of the system shown in figure 7. The difference between the two systems is the following. In the system in figure 20, a Stage-i kanban is released from a part after that part exits the manufacturing facility of Stage i. In the system in figure 7, on the other hand, a Stage-i kanban is released from a part after that part leaves the output buffer of Stage i. This is reflected in the initial distribution of free kanbans in the two systems. Namely, in the system in figure 20, Queue A_i initially has K_i free kanbans whereas in the system in figure 7 Queue A_i initially has $K_i - S_i$ free kanbans, since S_i kanbans are engaged onto an equal number of parts in Queue PA_i. The difference in the time when kanbans are released in the two systems, however, does not affect the timings of other events in the two systems; therefore the two systems are equivalent.

It can also be shown that when $K_i < S_i$, for all i, the behavior of the system shown in figure 20 is exactly the same as that of the system shown in figure 12. This is not so evident by comparing the two figures. In this paper, however, it suffices to say that it is shown in [7] that GKCS with $K_i < S_i$, for all i, is equivalent to the so-called Local Control System in which a part is released from the output buffer of a stage into the input buffer of the downstream stage if (1) such a part exists, (2) the input buffer and the manufacturing facility of the following stage are not full, and (3) the input buffer, the manufacturing, and the output buffer of the following stage, combined, are not full. It is not too difficult to see that these three conditions are the same as those in the system shown in figure 12.

Appendix D. Alternative queueing network model of the EKCS

The original queueing network model of the EKCS [11] is shown in figure 8 and really captures the essence of the EKCS: namely, that the EKCS is a superposition

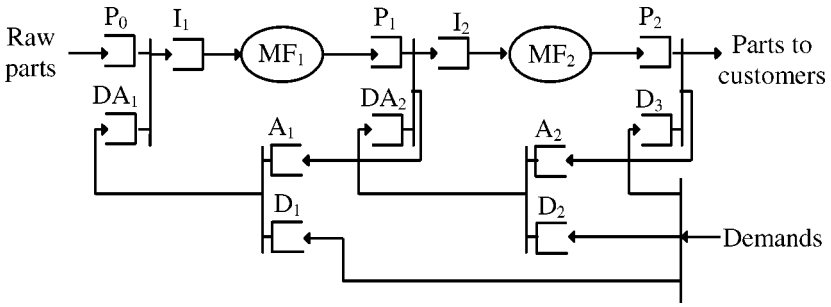

Figure 21. Alternative model of an EKCS with two stages in series.

of the BSCS and the KCS. An alternative queueing network model of the EKCS is shown in figure 21. The initial conditions in that network are the same as those in the GKCS shown in figure 7. It can be shown that the behavior of the system shown in figure 21 is exactly the same as that of the system shown in figure 8, once one notices that, in the system shown in figure 21, the synchronization station linking Queues A_i and D_i can be collapsed onto the synchronization station linking Queues P_{i-1} and DA_i, $i = 1, 2$.

The purpose of showing figure 21, which seems much more complicated than figure 8, is to better understand the difference between the GKCS and the EKCS. The difference between the GKCS, shown in figure 7, and the EKCS, shown in figure 21, is in the way customer demands arrive to Queue D_1. In the GKCS, a customer demand arrives to D_1 only after the synchronization station linking Queues A_2 and D_2 "fires", whereas in the EKCS a customer demand arrives to D_1, as soon as it arrives to the system.

References

[1] S. Axsäter and K. Rosling, Installation vs. echelon stock policies for multilevel inventory control, Management Science 39(10) (1993) 1274–1279.
[2] B.J. Berkley, A review of the kanban production control research literature, Production and Operations Management 1(4) (1992) 393–411.
[3] A.M. Bonvik, Performance analysis of manufacturing systems under hybrid control policies, Ph.D. thesis, Operations Research Center, Massachusetts Institute of Technology (1996).
[4] A.M. Bonvik, C.E. Couch and S.B. Gershwin, Comparison of production-line control mechanisms, International Journal of Production Research 35(3) (1997) 789–804.
[5] J.A. Buzacott, Queueing models of Kanban and MRP controlled production systems, Engineering Cost and Production Economics 17 (1989) 3–20.
[6] J.A. Buzacott and G.J. Shanthikumar, A general approach for coordinating production in multiple cell manufacturing systems, Production and Operations Management 1(1) (1992) 34–52.
[7] J.A. Buzacott and G.J. Shanthikumar, *Stochastic Models of Manufacturing Systems* (Prentice Hall, 1993).
[8] D.Y. Cheng and D.D. Yao, Tandem queues with general blocking: a unified model and comparison results, Journal of Discrete Event Dynamic Systems: Theory and Applications 2 (1993) 207–234.

[9] Y. Dallery and S.B. Gershwin, Manufacturing flow line systems: a review of models and analytical results, Queueing Systems 12 (1992) 3–94.
[10] Y. Dallery and G. Liberopoulos, A new kanban-type pull control mechanism for multi-stage manufacturing systems, in: *Proceedings of the 3rd European Control Conference* Vol. 4(2) (1995) pp. 3543–3548.
[11] Y. Dallery and G. Liberopoulos, Extended kanban control system: combining kanban and base stock, IIE Transactions 32 (2000), to appear.
[12] Y. Dallery, Z. Liu and D. Towsley, Properties of fork-join queueing networks with blocking under various operating mechanisms, IEEE Transactions on Robotics and Automation 13(4) (1997) 503–518.
[13] S.B. Gershwin, *Manufacturing Systems Engineering* (Prentice Hall, 1994).
[14] S.B. Gershwin, Design and operation of manufacturing systems: control- and systems-theoretical models and issues, in: *Proceedings of the 1997 Automatic Control Conference* (1997) pp. 1909–1913.
[15] W.J. Hopp and M.L. Spearman, *Factory Physics* (Irwin, 1996).
[16] F. Karaesmen and Y. Dallery, A performance comparison of pull-type control mechanisms for multi-stage manufacturing, International Journal of Production Economics 63(1) (2000) to appear.
[17] J. Kimemia and S.B. Gershwin, An algorithm for the computer control of a flexible manufacturing system, IIE Transactions 15(4) (1983) 353–362.
[18] D. Mitra and I. Mitrani, Control and coordination policies for systems with buffers, Performance Evaluation Review 17(1) (1989) 156–164.
[19] J.A. Muckstadt and S.R. Tayur, A comparison of alternative kanban control mechanisms, IIE Transactions 27 (1995) 140–161.
[20] M.L. Spearman, D.L. Woodruff and W.J. Hopp, CONWIP: a pull alternative to kanban, International Journal of Production Research 28(5) (1990) 879–894.
[21] M.H. Veach and L.M. Wein, Optimal control of a make-to-stock production system, Operations Research 42 (1994) 337–350.
[22] P. Zipkin, A kanban-like production control system: analysis of simple models, Working paper no. 89–1, Graduate School of Business, Columbia University, New York (1989).