9-30-1989

Manufacturing workstation model using Petri nets

Reni Alexander
New Jersey Institute of Technology

Abstract

Title of Thesis: **Manufacturing Workstation Model Using Petri Nets**

Reni Alexander,
Master of Science in Electrical Engineering,1989

Thesis directed by: ....
Prof. Anthony D.Robbi
Associate Professor
Department of Electrical Engineering

Petri nets have become a powerful tool for modeling and analyzing asynchronous concurrent systems. In this thesis I apply Petri nets for analyzing and modeling a flexible manufacturing workstation (FMS). A manufacturing workstation is modeled by using Colored- Timed Petri nets, with the consideration of tool handling, raw materials, quality checking and the reuse of reworkable scraps. A simulation program for a simplified Petri net model of manufacturing workstation is described and the production throughput is computed.

MANUFACTURING WORKSTATION MODEL USING PETRI NETs

by
Reni Alexander

Thesis submitted to the Faculty of the Graduate School of
the New Jersey Institute of Technology in the partial fulfillment of
the requirements for the degree of
Master of Science in Electrical Engineering
1989

Blank Page

APPROVAL SHEET

Title of Thesis: **Manufacturing Workstation Model Using Petri Nets**

Name of Candidate: **Reni Alexander**
Master of Science in Electrical Engineering
1989.

Thesis and Abstract Approved: 9/7/89
Prof. Anthony D.Robbi Date
Associate Professor
Department of Electrical Engineering

Thesis Committe

Prof.Constantine Manikopoulos
Date. 9/7/89...

....
Prof.Irving Wang
Date. 9/7/89

Vita

Name: Reni Alexander

Permanent Address:

Degree and date to be conferred: MSEE, 1989

Date of birth:

Place of birth:

Secondary education: St. Stephen's High School,1979.

Collegiate institutions	Dates	Degree	Date of Degree
New Jersey Institute of Technology, New Jersey.	1988-89	MSEE	October 1989
University of Tulsa Tulsa, Oklahoma.	1988	MSEE	
University of Mysore, Mysore Karnataka, India	1982-87	B.E	May 1987
University of Kerala Kerala, India	1979-81	PDC	Augest 1981

Major: Electrical Engineering.

Dedicated to
Appachan, Ammachi and Ammachi

Acknowledgement

The author wishes to thank his thesis advisor Prof. Anthony D.Robbi for providing much time, effort, help and advice through out his time at New Jersey Institute of Technology.

The author also wishes to thank his thesis committe for their help and advice at New Jersey Institute of Technology.

The author wishes to thank his family for their love and understanding through out his studies.

Finally author wishes to thanks the members of the Electrical Engineering Department at New Jersey Institute of Technology for their support.

Contents

List of Figures

List of Tables

Chapter 1

INTRODUCTION TO FLEXIBLE MANUFACTURING SYSTEM (FMS)

1.1 Flexible Manufacturing Systems

A Flexible manufacturing system (FMS) is an integrated, computer controlled configuration of machine tools and automated material handling devices that simultaneously process medium sized volumes of a variety of parts types. High productivity is achieved in such system by effectively incorporating principles of group technology, total quality control, etc., and following production control strategies such as manufacturing resource planning and just-in-time production.

Flexible manufacturing systems are discrete event dynamical systems in which the workpieces of various job classes enter the system asynchronously and are processed concurrently, sharing the limited resources, viz, workstation, robots, buffer and so on. In addition the system to be evaluated may machine many different kinds of workpieces which require different sequence of operations with significantly different machining times. Another feature of the FMS is a concurrency which means that each acting element may move independently or require synchronism in part.

1.2 Models for FMSs

The main models used to date to analyse performance of FMSs have been queueing networks, perturbation analysis and simulation. Queueing network analysis provides information about the average system behavior observed over a long time period (steady state). It is useful for qualitative guides for and planning problems of the FMS. Perturbation analysis "views a queueing network as a stochastic dynamical system evolving in time and observes the simple realization of its trajectory", similar to simulation. However by perturbing one event, and following through an analysis, many useful questions can be answered without repeating a simulation run. Perturbation analysis has been used to answer questions that a product form queueing network can not. Most often real time issues are analyzed using simulation , which is flexible, but time consuming and expensive. Some other features of the above modeling techniqus are explained next.

1.2.1 Queueing Networks

Queueing network models take into account system dynamics, interactions and uncertainties inherent in manufacturing systems. Also efficient computational algorithms are available for solving the queueing network model.

Suri and Hildebrant have proposed to view the manufacturing system as a closed queueing network with multiple customer classes and solved the design and real time operation problems using the mean value analysis method [21].

1.2.2 Simulation

The modeling and performance analysis of a manufacturing system can be conducted using discrete event simulation[15]. Such a simulation views the system operation as a succession of events, and in principle, can mimic system behavior in as much detail as desired. However performance optimization with respect to a number of decision parameters requires a large number of simulation runs. Also because of the randomness involved in the FMS operations such as breakdown, content of a job queue etc., multiple simulation runs and subsequent output analyses are required to reduce the simulation error.

1.2.3 Perturbation Analysis

Perturbation analysis is another important technique useful in the performance analysis of FMSs. Using the data obtained from simulation, perturbation analysis method finds the sensitivity of the system performance measures to changes in decision parameters. For the simulation studies of N-parameters, it provides an N-fold increase in efficiency over the "brute force" simulation method. Perturbation analysis typically answers questions such as " what would be the production of all types today if there were one more fixture of part type A"?

1.3 Basic Petri Nets

I intend to use Petri nets to model the flexible manufacturing system in this thesis. Place-Transition nets are special bipartite graphs. They are also known as Petri nets. An example Petri net is shown in Figure 1-1. The standard Petri net model is defined by a set of places, a set of transitions and a set of directed arcs which connect places to transitions or vice verse. Places are represented by circles and transitions by bars. Places may contain tokens (drawn as dots). A Petri net with tokens is a marked Petri net. The marking of a marked Petri net is a vector, the elements of which are given by the distribution of tokens in the places of the net.

A marking represents the state of system being modeled. Generally places represent conditions and transitions represent events. A place is an input (output) place of a transition if an arc exists from the place (transition) to the transition (place).

The dynamic behavior of a system is modeled as follows: the occurrence of an event (state change) is represented by the firing of the corresponding transition. The movement of tokens in the net resulting from the firing of one or more transitions represents a change in the state[12,13].

1.3.1 Advantages of modeling a system using Petri nets

1. They describe the modeled system graphically and hence enable an easy visualition of complex systems.

2. Petri nets can model a system hierarchially; a system can be represented in a top-down fashion at various levels of abstraction and detail [15].

3. A systematic and complete qualitative analysis of the system is possible by well-developed Petri net analysis techniques [13].

4 The existence of well-formulated schemes for Petri net synthesis facilitates system design and synthesis [18].

5. Performance evaluation of systems is possible using timed Petri nets [3,9,17].

6. Concurrency and synchronism can be modeled.

1.3.2 Modeling principles for applying Petri net to a FMS

The following modeling conventions are suggested to allow the application of Petri nets to FMS systems.

1. The set of activities (tasks, operations) to be performed by the production system is represented by a set of transitions. Each transition in this set is assigned a corresponding activity.

2. An activity is defined by the simultaneous use of one or more resources. In particular, the input places of a particular transition indicate the resources which the associated event simultaneously requires. Output places serve as resources for activities that are required next.

3. The activity tokens of various types make available resources at hand.

1.3.3 Properties of Petri nets useful in modeling

In the Petri net model unrelated events can occur independently and in effect simultaneously. Another major feature of Petri net is their asynchronous nature. There is no inherent measure of time or flow of the time in a Petri net except for timed transition. The Petri net structure itself contains all necessary information to define the possible sequences of the events of the modeled system.

A Petri net can reflect a real-time situation where several things are happening concurrently. The order of occurrence of events is not unique so that any of a set of sequences may occur. While non-determinism is advantageous from the modeling point of view, it introduces considerable complexity in the analysis of Petri net. To reduce this complexity, one limitation is generally accepted in the modeling of systems by Petri net: the firing of a transition is considered to be instantaneous. The exception is the aforementioneed timed transition.

Petri nets are also used for analyzing synchronizing nature events, like the dining Philosophers problem explained by J.L.Petrson [13].

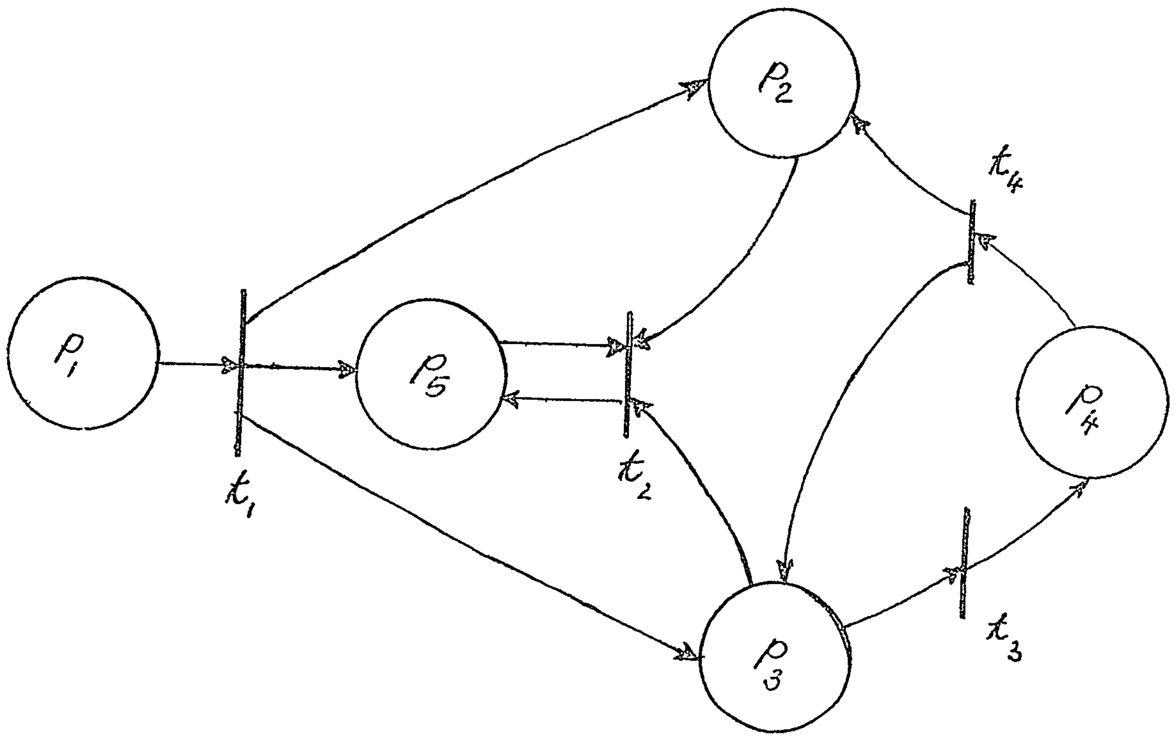


Figure 1.1: Simple Petri Net

Chapter 2

DESIGN AND PERFORMANCE OF FMS

2.1 Design of Flexible Manufacturing systems

An FMS basically consists of a set of machine tools interconnected by an automated palletized transportation system. The main advantages of an FMS are the ability of working different types of parts and, as consequence, of adapting to changes in production mix and volume rapidly, and also the capability of ensuring a lower but continuous production in the case of accidental or programmed machine stops. The components of an FMS are single purpose machine tools, such as lathes, boring and milling machine, or multi-purpose machine tools such as a machining center. For each operation each machine requires the proper tool or proper set of tools.

Computer control allows a plant to operate automatically by assigning operations to machines and by directing the delivery of proper tools and routing of parts. Dedicated fixtures are used to accommodate parts on pallets (buffer). An FMS architecture not only makes it possible but also requires that a relatively high number of properly matched parts be contemporaneously present in the system in order to obtain a balanced utilization level of the resources, machines, tools and fixtures.

The use of flexible manufacturing systems has led to a marked increase in productivity and reduction in inventory costs of industries that manufacture a set of related parts at low to medium volumes.

Modeling, analysis and performance evaluation studies of FMSs are of immense practical interest to establish feasibility, evaluate qualitative and quantitative performance and compare alternative FMS configurations.

In the design and construction of FMS, a performance evaluation of the system is required at two different stages of development: At the beginning of the design of the system when the machines, the layout, and the transport system are chosen and dimensioned. At this stage the performance estimates need not be particularly accurate; usually only the order of magnitude of the parameters of the system are known in this initial phase, and crudely approximated analytical methods are used [4].

During the tuning phase of the system, where most of the parameters (the machines, lay out, etc) have already been chosen, and only minor changes in the system (e.g. the size of the buffers, or production mixes) can be modified in order to optimize the overall throughput or the machine utilizations. For this purpose an accurate performance evaluation of the system is required. It is usually obtained by developing detailed simulation models.

2.2 Application of Petri net to FMSs

Petri nets are described as an ideal tool to model the complex interactions among different processes in the FMSs. A preliminary investigation of the use of timed Petri nets in the study of real time control and performance evaluation of FMSs is carried out by Dubois et al [6]. Also manufacturing systems that can be modeled by a restricted class of Petri nets (safe, decision free, and strongly connected PTNs) are analyzed and simple performance evaluation algorithm, based on a special algebraic structure is presented for the restricted class of Petri net models [15].

The power of Petri nets lies in their ability to prove certain qualitative properties of systems. Invariant analysis is often used to prove the properties of Petri net models. It is in this respect that the recent result [18] for fast and easy computations of the invariants assumes much significance. The main result developed make it possible to determine the invariants of the union of two Petri nets when the invariants of the individual Petri nets are known. A Petri net model of a given FMS is synthesized by constructing the union of simpler Petri net models that correspond to certain primitive operations into which the given FMS structure can be decomposed. At each stage of the synthesis, the properties of Petri nets are evaluated; analysis and synthesis proceed in parallel.

I intend to use different simpler Petri net units to constructs the final Petri net model of the manufacturing workstation.

Chapter 3

MARKED-TIMED PETRI NETS

3.1 Rules of Marked Petri Nets

Following are the rules of marked Petri nets:

1. A transition is enabled when each of its input places contains at least one token.

2. A transition can fire only if it is enabled.

3. When a transition fires a token is removed from each of its input places, and a token is deposited into each of its output places.

A marking M1 is immediately reachable from a marking M if we can fire some enabled transition in marking M resulting in the marking M1.The reachability set of a marked Petri net is the set of all markings that are reachable from a given initial marking.

3.2 Reachability set of a Petri net

From a marking M, a set of transition firings is possible. The result of firing a transition in a marking M is a new marking M1. We say that M1 is immediately reachable from M if we can fire some enabled transition in the marking M resulting in the marking M1. A marking M1 is reachable from M if it is immediately reachable from M or is reachable reachable from any marking which is immediately reachable from M.

The reachability set of a marked Petri net is the set of all states in to which the Petri net can enter by any possible execution.

3.3 State space of Petri net

The state of a Petri net is defined by its marking. Thus the firing of a transition represents a change in the state of the net. The state space of a Petri net with n places is the set of all markings.

The change in state caused by firing a transition is defined by a partial function d, called the next state function. Given a Petri net and an intial marking M0, we can execute the Petri net by successive transition firings. Firing a transition tj in the intial marking produces a new marking M1 = d (M0,tj). In this new marking, we can fire any new enabled transition, say tk, resulting in a new marking M2 = d(M1,tk). This can continue as long as there is at least one enabled transition in each marking. If we reach a marking in which no transition is enabled, then no transition can fire and the execution of the Petri net must stop.

3.4 Colored Petri nets (CPNs)

A colored Petri net is a generalization of Petri net in which a color is associated with each token, giving it an identity, a specific set of attibutes [15]. Also a set of colors is associated with each place and transition. The set of colors associated with a place indicates the color of the tokens that can be present at the place. When a transition is fired, tokens are removed and added at its input and output places respectively, as in a marked Petri net. In addition, functional dependency is specified between the color of the transition and the color of the involved tokens. The color attached to a token involved in a transition firing may be changed. It often represents a complex information unit.

Color Petri nets have the same modeling power as the basic Petri nets. The main advantage of colored Petri net over Petri net is the possibility of getting a more compact representation of a large and complex system. In an FMSs, for example, color may distinguish different tool types, jobs etc.

3.4.1 Main feactures of colored Petri nets

1. Each token is assigned a color and which stands for a set of attributes.

2. Each color represents a particular type; there is also a neutral type.

3. A transition is enabled if all its input places contain token with equal colors; a neutral color matches any other color. When a transition fires, it removes tokens from the input place and deposits token in to the output place setting their colors and states according to the inscriptions on the outgoing arcs. Output colors and states can be functions of input ones [9].

3.5 Timed Petri nets

The standard Petri net usually omits the concept of time as a parameter. Time has been added to Petri net model in several ways. The timed Petri net (TPN) uses a fixed number of discrete time intervals, the stochastic Petri net (SPN) uses an exponentially distributed random variable. The discrete time SPN fills the gap between the TPN and the normal SPN [16].

The operation of a Timed Petri net may be thought of as follows. When a transition is enabled, remove the enabling input tokens; later when it fires place output tokens. In an alternate approch, used in this thesis, it is assumed that tokens remain in the input places until the timed transition fires.

A timed Petri net works as follows: when a transition is enabled, establish a time interval [a,b] measured from enabling, when it may fire, provided that the required token are still present at the end of the interval. This allows the presence of tokens to be tested as a transition firing is in progress, a useful feature. In case of conflict this is a first ready, first served queue rather than a first come, first served queue of the timed net.

Delay times associated with transitions may be fixed, stochastic or deterministic. The allowance of finite firing times makes certain computations on a Petri net difficult. For example the reachability set (set of markings which can be reached from a given initial marking) may be difficult to compute.

Conflict occurs when more than one transition on the output arcs of a given places are enabled and firing one would disable the other. Which transition should fire? while it may seem desirable to aviod such constructs, they model real situations such as resource shearing. The following rules can be applied in some cases: instantaneous transition are enabled at the expense of timed ones [2].

3.6 Inhibit Arcs

An inhibit arc to a transition represents that the transition cannot fire when there is a token in that particular place. Inhibition function is represented with a small-circle headed arc directed towards a transition from any place contained in its inhibitation.

I intend to use Petri net comprised of a set of places, a set of neutral and color tokens, a set of instantaneous and timed transitions and three function defined on the set of transition (input, output and inhibition function). All the above features makes the Petri net a good tool for modeling an FMSs.

Chapter 4

PETRI NET FORM OF MANUFACTURING WORKSTATION

4.1 Previous Model

4.1.1 Overview

Modeling of a manufacturing workstation (system) using Petri net has been attempted by different people in different ways. One of them is explained below.

A Petri net model of a manufacturing workstation by Prof. Anthony D. Robbi of New Jersey Institute of Technology is shown in Fig. 4-1 [2]. Robbi's model includes timed transitions. This is a basic model for my variant and is explained below.

Q1 is a queue for incoming jobs and it feeds a jobtoken (color token in a sense) into the net under control of a dispatcher. There is a one job staging place, NEXT, which contains the next job to be processed. It moves into processing after a finite setup time represented in the transition SETUP. That setup time is overlappable with the processing of previous jobs. Nonoverlappable setup must be included in the processing time itself. It may depend on the nature of the preceding job; the sub-net comprising NEXT, TEST, MEMORY, NORMAL, READY, and FLUSH arranges for a copy of the previous job token to reside in MEMORY. When a jobtoken arrives in NEXT. At this time the setup attribute of the jobtoken in NEXT is compared to the setup requirements to the jobtoken in MEMORY. Difference can be accounted for by adjusting the setup time and processing times of the incoming job. The subnet between SPLIT and the two alternative processing sites, PSCRAP and POK allows for random machine failure and downtime with variable consequences to the work in process (WIP). Processed work flows through the timed transition MS and MG to the output staging location EXIT. The quality attribute of the finished work through MG is computed at the time.

The time a job (colored token) spends in the system depends in part on the delay caused by the previous job.

4.1.2 Places

The following places are utilized for jobtokens:

NEXT	hold the next jobtoken for comparison with last and for overlappable setup time.
TEST	temporarily holds job to drive logic and wait for previous job to complete setup and processing.
SPLIT	temporarily holds job to allow probabilistic subnet to steer job in to one of the two processing regimes.
PSCRAP	holds job for processing under machine failure situation, quality is scrap.
POK	holds job for normal processing, quality variable.
EXIT	collects the finished work.
MEMORY	holds the previous jobtoken.

The following places are used for control tokens:

NORMAL	waits for the job to emerge from NEXT.
READY	waits for the job to emerge from Q1.
FLUSH	restarts control cycle by emptying MEMORY.
MUP	token present indicate normal processing.
MDOWN	token present indicates processing suspended.

MSCRAP output quality scrap.

MGOOD output quality normal range.

4.1.3 Transitions

The minimal set of action is shown on the outgoing arcs of some transition:$<n>$ converts a jobtoken to nutral type, and $< qual - -var >$ indicates that a quality attribute has been assigned to a jobtoken passing through. An open rectangle transition may take measurable time.

T1 fires when NEXT and TEST are empty and a jobtoken in Q1.

T2 advances jobtoken after T8 has fired.

T3 or T4 fires to advance jobtoken to appropriate process.

T5 moves completed work to Q0.

SETUP enabled after jobtoken enters NEXT, delay time computed using comparison of color attribute with color of token in MEMORY.

MS or MG enable after job moves to processing site, delay time is a function of the processing mode and recipe attached to the jobtoken.

TD represents time processing is suspended.

TU represents time in processing mode.

TS represents relative time failed that machine makes scrap.

TG represents relative time failed machine makes good product probability of good is TG/(TG + TF)

4.2 New Model

4.2.1 Modeling of Manufacturing Workstation

My model is a variant of the model explained above with the consideration of tools, raw materials, other consumable parts, quality checking and the reuse of reworkable products.

The schematic diagram of the manufacturing workstation is shown in Figure 4-2. Which shows a queue of jobs arriving in the workstation. Each job waits for the completion of previous once in the workstation. Raw materials, consumable parts and tools are fed to the work station. After processing the tools are fed back to a tool room. Finished products are inspected for their quality. If the product is good it is then moved for dispatching and shipping, otherwise it is inspected for the chance of rework and fed back to the workstation if possible. Otherwise get the reusable parts and throw out the garbage.

A Petri net model of a manufacturing workstation with the consideration of tools, raw materials, consumable parts, quality checking and the reuse of reworkable products is shown in Figure 4-3.

Q1 is a queue of incoming jobs and it feeds jobtoken (color token) into INITIAL by firing the transition T1 when INITIAL is empty. Jobtoken is fed to a job staging place, NEXT, which contains the next job to be processed by firing the transition T2 when TEST and NEXT are empty. By firing the transition we get three jobtokens in NEXT. The timed transition SETUP moves one token to TEST and another to MEMORY after a finite setup time. That time may be overlappable with the processing of the previous job. Non-overlappable setup must be included in the processing time itself, depending up on the nature of the present and previous jobs. The other two tokens in NEXT are used to select the raw materials and tools for the production. Raw materials and other consumable parts are moved to STORE from RAWP by firing the timed transition TT1.Tools are moved to TOOL either from the INV by firing TT2 or from FREE by firing transitions T10 and T12.

In order to avoid a dead lock situation in firing the transitions from NEXT it is assumed that the firing time of transition TT2 is greater than that of transitions TT1 and SETUP. To explain the above situation let us consider FIgure 4-4. If the previous job in MEMORY and the present job are the same the transition T10 fires and allows the tool token at FREE to to go TOOL by firing the transition T12, otherwise not. Since we assume that firing time of TT2 greater than that of TT1 and SETUP, TT2 will fire only after TT1 and SETUP. So if the present job is same as the previous job TT2 dose not fire.Otherwise TT2 fires and moves the tool token from INV to TOOL, simultaneously the timed transition DELAY fires and moves the token from MEMORY to DIFF if TEST is empty. Then by firing the transition T11 a tool token at FREE is moved to the INV. It is assumed that the time taken to fire DELAY transition is less than that of SETUP transition when the present and previous jobs are not matched in order to avoid multiple tokens at MEMORY.

The job token in TEST is moved to PROD when T3 fires. T3 can fire only when the previous job clears PROD, the proper color tokens are in TOOL and STORE, and there is no control token in DOWN. A control token in DOWN represents the inoperable state of the workstation.

The jobtoken in PROD is then processed and moves to SPLIT through the timed transition PROCESS. The product token in SPLIT goes to one of two destructions FINAL (qualified product) and SCRAP. This is done by imposing the quality randomly through the circulating token in GOOD and BAD.

The good product (FINAL) is then dispatched and transported to outside world through the timed transition TT3.

SCRAP is checked for the possibility of rework by the nature of the control token in RY-RN subnet. The reworkable SCRAP is fed back to INITIAL through REWORK and timed transition TT4. TT4 can fire only when there is no token at INITIAL. Otherwise the SCRAP is checked for usable parts.

The position of the control token in the PY-PN subnet determines this. The usable parts are fed back to the RAWP through GPART and timed transition TT5. Else the garbage (GARB) is thrown out.

The above process is repeated again and again as jobs flow through the cell.

4.2.2 Places

The following places are utilized for jobtoken:

Q1	Queue of incoming jobs.
INITIAL	Holds the rework and new jobtokens.
NEXT	Holds the next job token for comparison with previous jobtoken, for overlappable setup time and to chose proper tools and raw materials for the production.
TEST	Temporarily holds job to drive logic and wait for previous job to complete processing.
PROD	Production place, it holds the job for processing.
SPLIT	Temporarily holds the product to allow probabilistic subnet to check the quality of the products.
SCRAP	Temporarily holds the scrap to allow probabilistic subnet to some time cause rework.
RSCRAP	Temporarily holds the product to allow probabilistic

subnet to extract good parts from the scrap.

FINAL	Holds the product before dispatching and transporting to out side world.
MEMORY	Holds the previous jobtoken to compare with next.
REWORK	Holds the reworkable unit before feeding back to the system.
GPART	Holds the reusable parts before feeding back to RAWP.
GARB	Holds the garbage before throwing it out.
Q0	It represents the out side world.
INV	Holds the tool before transporting to the TOOL.
TOOL	Holds the tool before moving to the production line.
RAWP	Holds the raw materials and other consumable parts before transporting to the STORE.
STORE	Holds the raw materials and other consumable parts for the production.
FREE	Holds the previous tools before moving to either to INV or to TOOL.
SAME	Holds the token if the present and previous jobs are the same.
DIFF	Holds a token if the present and the previous jobs are different.

The following places are utilized for control tokens:

UP	Token present indicates normal processing.

DOWN	Presence of token indicates processing is suspended.
GOOD	Output quality in normal range.
BAD	Output quality is scrap.
RN	Scrap cannot rework.
RY	Scrap can rework.
PN	Absence of reusable parts in scrap.
PY	Presence of reusable parts in scrap.

4.2.3 Transitions

T1	Fires when INITIAL is empty and a jobtoken in Q1.
T2	Fires when NEXT and TEST are empty and a jobtoken in INITIAL.
T3	Fires when PROD and DOWN are empty, jobtoken in TEST, and proper tools and raw material tokens in TOOL and STORE respectively.
T4, T5	Fire simultaneously with respect to the quality of the product and moves a token to either in FINAL or in SCRAP.
T6, T7	Fire simultaneously by the nature of the scrap and moves a token to either in REWORK or in RSCRAP.
T8, T9	Fire simultaneously by the nature of real scrap.

T10	Fires when present and previous jobs are same.
T11	Fires when tokens present in DIFF and FREE.
T12	Fires when tokens present in SAME and FREE.
SETUP	Enable after jobtoken enters NEXT, delay time computed using comparison of color attribute with color of token in MEMORY.
PROCESS	Enable after job moves to processing site, delay time is a function of processing mode and the attributes of the jobtoken.
DELAY	Enable when there is no token in TEST and there is a token in MEMORY.
TT1	Represents the transportation between RAWP and STORE depending up on the nature of job at NEXT.
TT2	Represents the transportation between INV and TOOL with respect to the nature of job at NEXT.
TT3	Represents the transportation and dispatching of the finished products.
TT4	Represents the transportation of reworkable unit to INITIAL.
TT5	Represents the transportation the reusable parts to RAWP.
TD	Represents time processing suspended.
TU	Represents time in processing mode.
TB	Represent the relative failed machine makes scrap.

TG	Represents the relative time machine makes good products.
TN	Represents the relative time that scrap cannot rework.
TY	Represents the relative time that scrap can rework.
TPN	Represents the relative time that parts cannot be used again.
TPY	Represents the relative time that parts can use again.

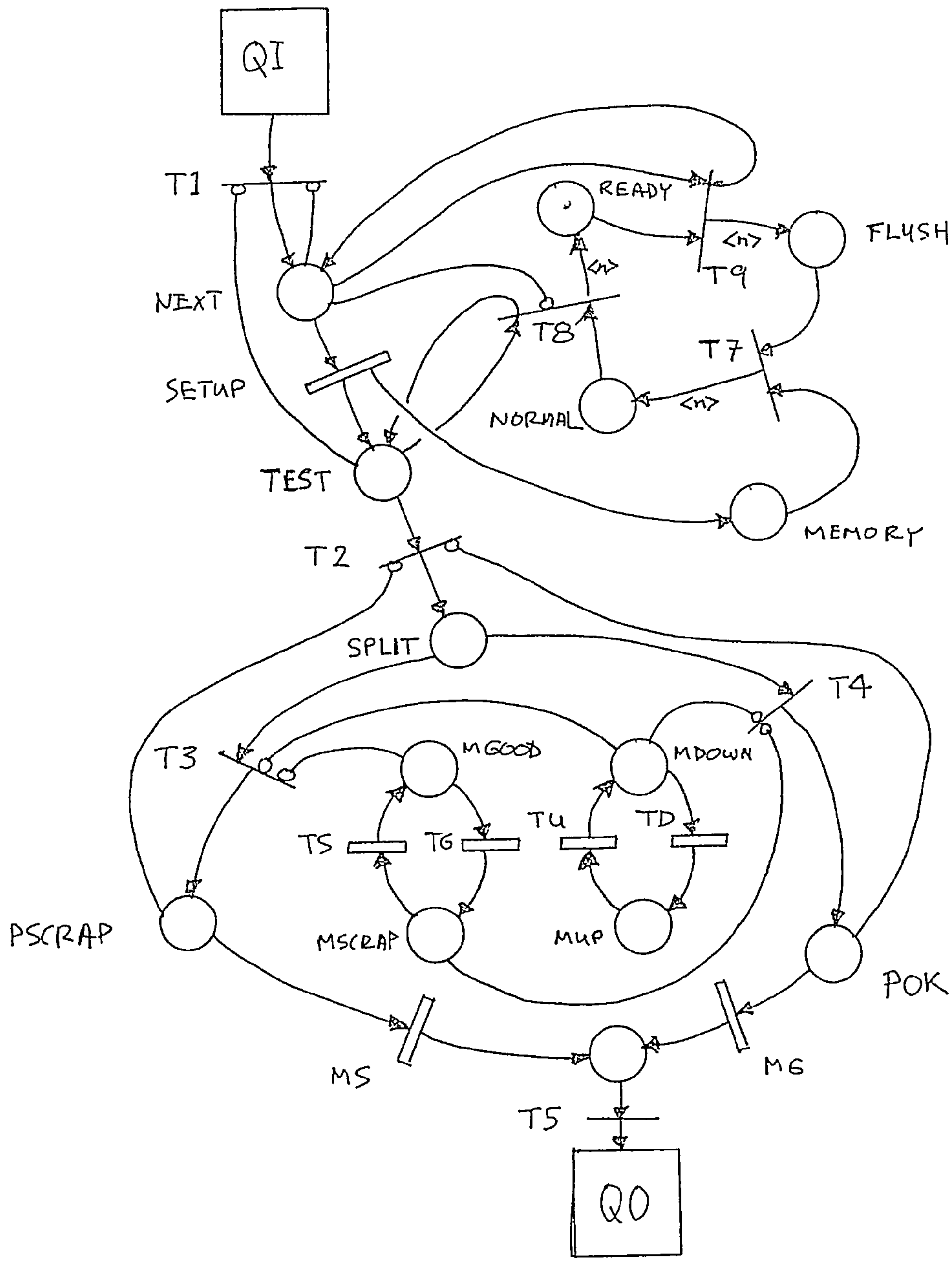


Figure 4.1: Petri Net Model of Factory Workstation

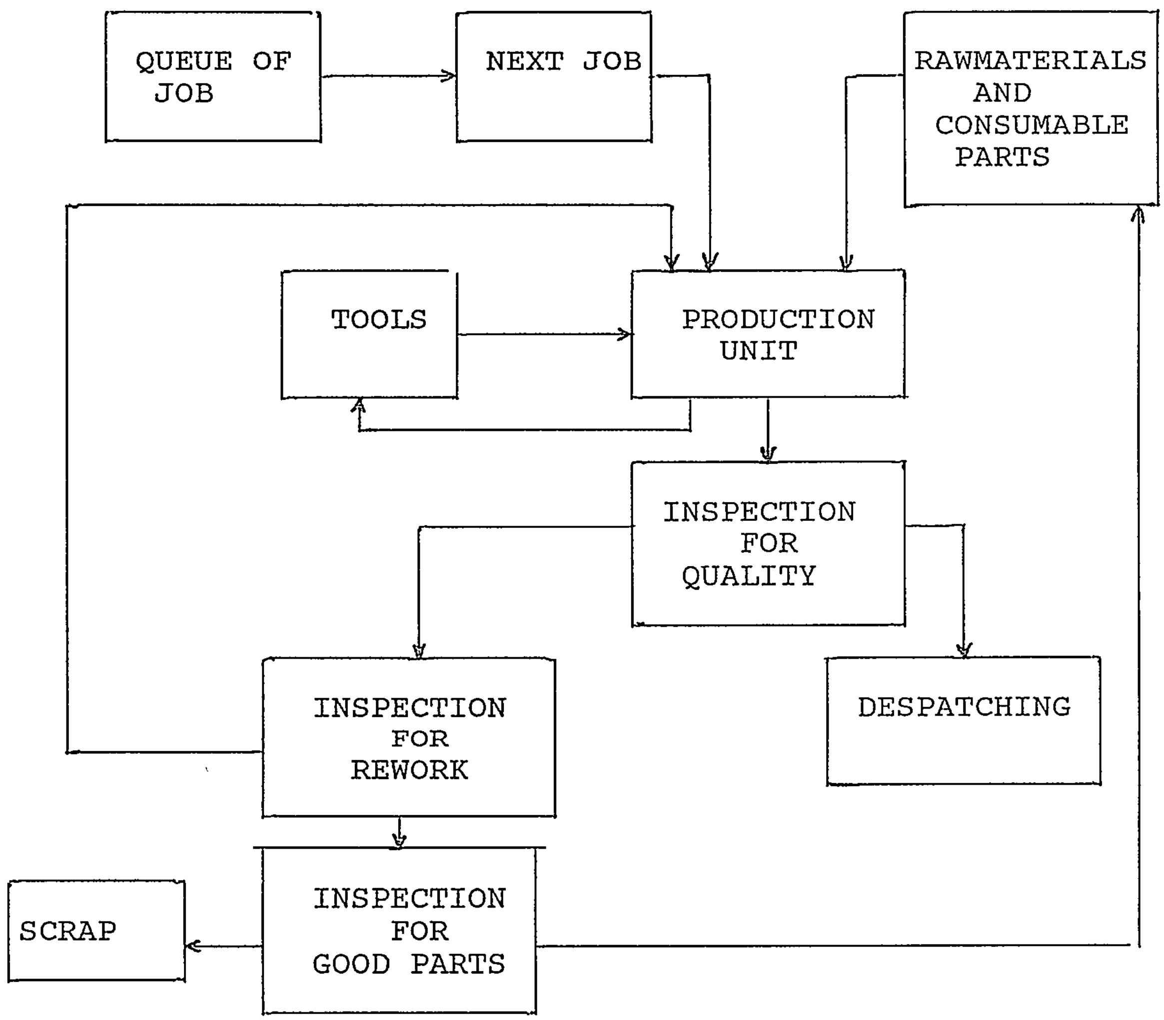


Figure 4.2: Schematic Diagram of Manufacturing Workstation

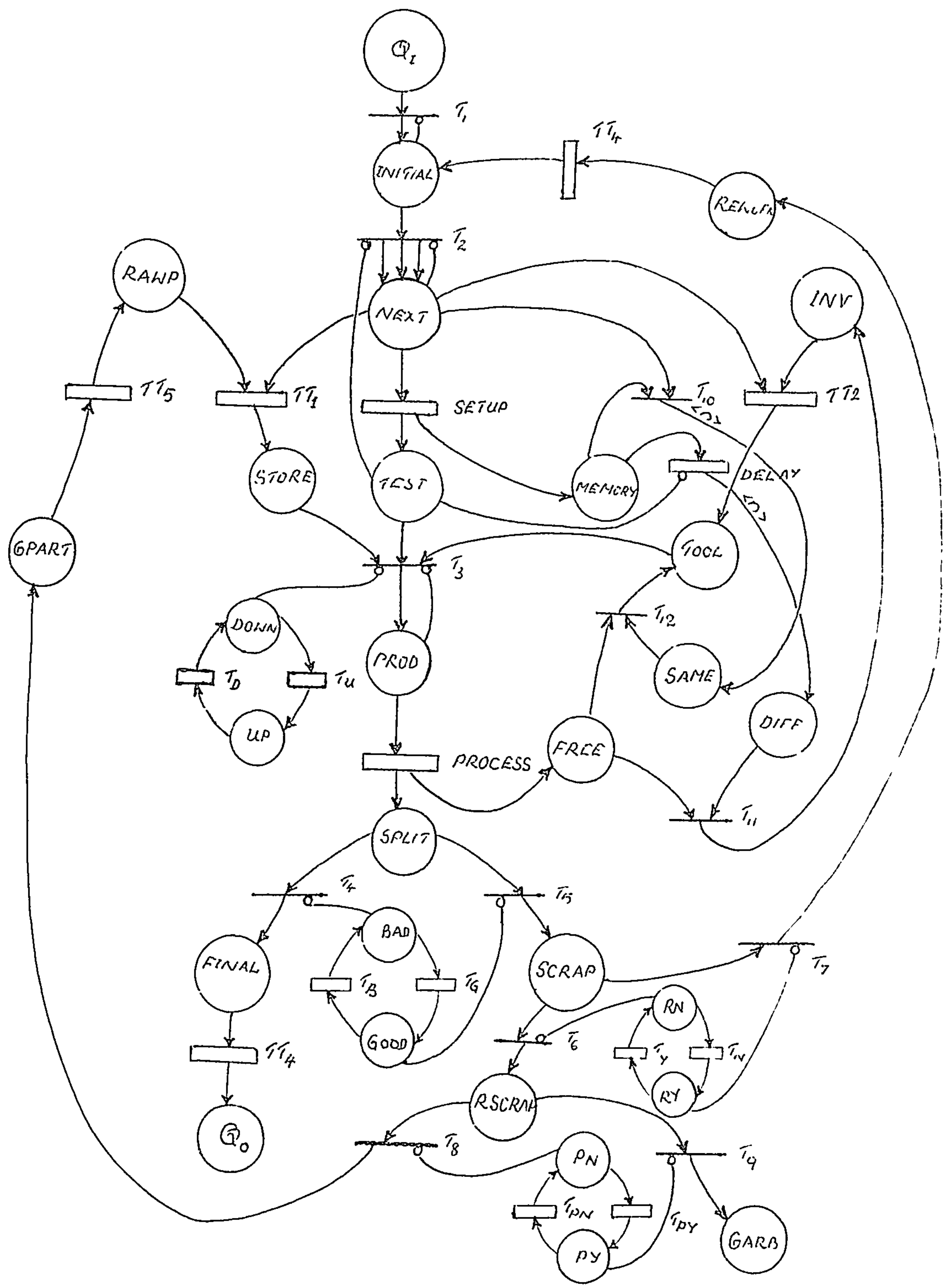


Figure 4.3: Petri Net Model of Manufacturing Workstation

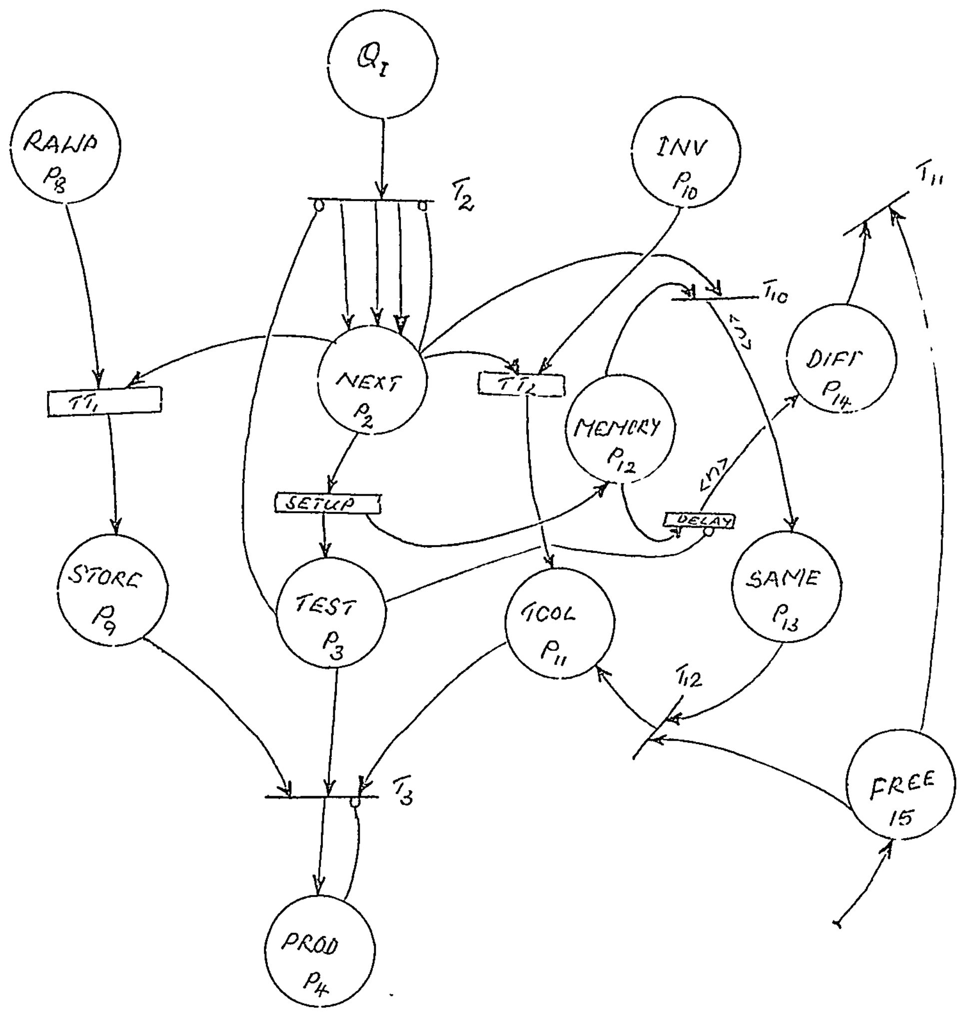


Figure 4.4: Potential Dead Lock Situation of the Petri Net Model

4.3 Operation of the FMS Petri Net

Consider the simplified Petri net model for manufacturing workstation shown in Figure 4-5.

The assumptions made for simplifying the Petri net model shown in Figure 4-3 are the following.

1. There is no processing breakdown,
2. SCRAP is not good for rework,
3. There are no useful consumable parts in the SCRAP.

Now consider the table 4-1, which shows the firing of a transition at each step as well as the working (except for the stochastic subnet GOOD/BAD).

Intially, there is a queue of job tokens at place P1 (QI),color tokens for raw materials at place P8 (RAWP) and color token for tools at P10 (INV).

First, T1 fires and places three tokens at NEXT.

Next, the timed transitions SETUP, TT1 and TT2 fire. A job token moves to TEST and another to MEMORY, a raw materials token moves to STORE and a tool token to TOOL. It is assumed that the firing time of the transition TT2 is greater than that of TT1 and SETUP.

Then, the transition T2 fires and moves the job token to PROD.

Next, transition T1 fires since TEST and NEXT are empty and moves the job token to NEXT. The timed transition DELAY starts counting for firing. The job token in PROD moves for processing and the timed transition PROCESS fires and moves the product token to SPLIT and tool token to FREE.

Then, Transition T5 fires if the previous job token in MEMORY and the present job token in NEXT are the same in processing requirementsand moves the token from MEMORY to SAME. Otherwise the DELAY transition fires and moves the token in MEMORY to DIFF, simultaneously the timed transition TT2 fires and moves a tool token from INV to TOOL. The timed transition SETUP fires and moves one token to TEST and one to MEMORY. The firing time of SETUP depends up on the nature of the job token in NEXT. The timed transition TT1 fires and moves a raw material token to STORE from RAWP.

Either transition T3 or T4 fires depending up on the quality of the product which is determined by the position of the control token in the GOOD/BAD subnet. If the product is good then T3 fires and moves the product token to FINAL for dispatching and shipping. Otherwise T4 fires and move the product token to SCRAP.

Either T7 or T6 fires depending up on the availability of token in SAME and

DIFF. If there is a token in SAME ,ie, next job is same as the present job, then T7 fires and moves the tool token from FREE to TOOL. Otherwise T6 fires and moves the tool token from FREE ti INV.

Afterwords, the process is repeated from step 3 and the cycle is reproduced again and again.

where it is?

F time	F Transi.	input place	Ih place	out place	No. token
0	T1	P1	P2,P3	P2	3
2	TT1	P2,P8		P9	1
3	SETUP	P2		P3,P12	1,1
4	TT2	P2,P10		P11	1
4	T2	P3,P9,P11	P4	P4	1
4	T1	P1	P2,P3	P2	3
9	PROCESS	P4		P5,P15	1,1
9	T5	P2,P12		P13	1
9	T3	P5	P17	P6	1
9	T4	P5	P16	P7	1
9	T6	P14,P15		P10	1
9	T7	P13,P15		P11	1
11	TT1	P2,P8		P9	1
12	DELAY	P12	P3	P14	1
12	SETUP	P2		P3,P12	1,1
13	TT2	P2,P10		P11	1
13	T2	P3,P9,P11	P4	P4	1

Table 4.1: Firing of Transitions in Each Step

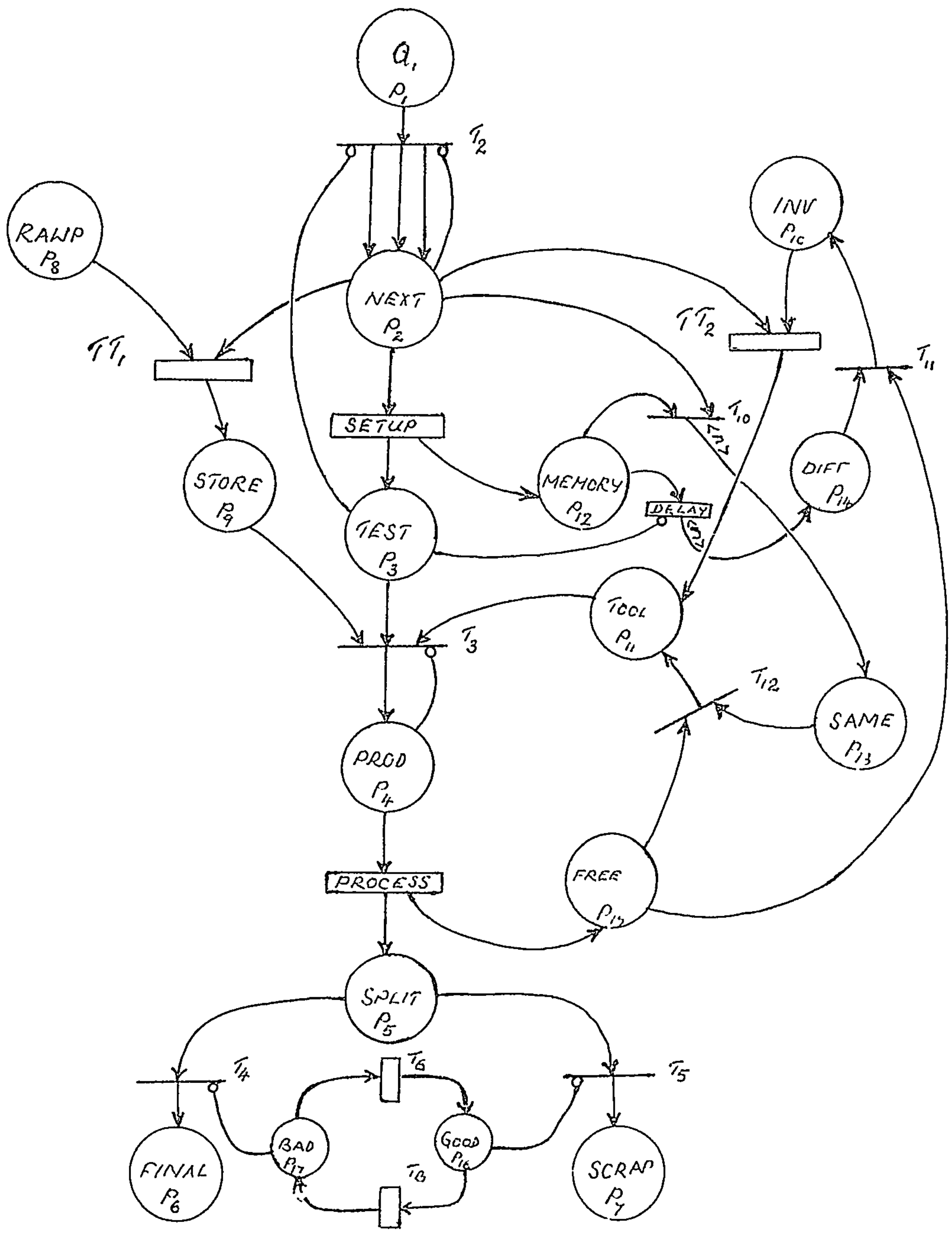


Figure 4.5: Simplified Petri Net Model of Manufacturing Workstation

Chapter 5

DYNAMIC SIMULATION OF THE PETRI NET FMS MODEL

5.1 Simulation Program

1. An event list containing the future occurrence of transition firings is maintained. The simulation clock is updated when a timed transition taken from event list is fired. Otherwise, it dose not advance.

2.Simulation Principles:

a) Initialize the value of the variables to reflect the initial marking of Petri net model. Initialize the clock.

b) Establish a transition list containing transitions enabled by the current marking (putting 1 in the transition lists for the enabled transition and 0 for others).

c) After firing an enabled transition the transition list is recomputed. Different transition lists are used for instantaneous and timed transitions.

d) New markings of Petri net model is accomplished by multiplying the enabled transition list with input/output matrix and adding it with the last marking. Input/output matrix contains the input output places of all transitions by indicating -1 for input and 1 for output.

e) When ever a token arrives in the output place (FINAL) the corresponding elapsed time and number of tokens in the output place(P6) are printed out.

f) For convenience it is assumed that same kind of job is appearing for the production, i.e. tokens are not colored.

Flow chart and the simulation program in basic for the simplified Petri net model of the manufacturing workstation is shown in appendix A.

Functions of matrices used in the program are follows:

Matrix M(I) represents the marking (tokens) in each places of the Petri net.

Matrix X(J) represents the lists of instantaneous transitions that are ready for firing.

Matrix Y(K) represents the lists of timed transition that can fire.

Matrix C(J,I) represents the input and output places for the instantaneous transition .-1 in the matrix represent the place is an input place, 1 represent the place is an output place and 0 represent there is no connection between that place and that particular transition.

Matrix D(K,I) represents the input output place for the timed transition.

Matrix E(J,I) represents the input inhibited arcs for the instantaneous transition. 1 in the matrix represents the place is an inhibited place and 0 represents there is no connection between that particular place and that particular transition.

Matrix F(K,I) represents the input inhibited arc for the timed transition.

Matrix Z(J) represents the fired instantaneous transition.

Matrix W(K) represents the fired timed transition.

5.2 Result

For the transition firing times shown in Table 4-1, the elapsed time, the number of output tokens in the output place (FINAL) and the expected time for three input tokens are shown in the Table 5-1.

Elapsed Time Unit	No. Token in FINAL	Expected Time Unit
9	1	9
17	2	17
25	3	25

Table 5.1: Comparison of Elapsed Time and Expected Time

The Petri net model of the manufacturing workstation is successfully implemented for the production of same natured jobs. The elapsed time unit for the production of each job exactly matches the expected time. Hence the throughput of the system is in the desired range.

5.2.1 Effects of timed transitions on throughput of the system

An increase in firing time of the timed transition TT1 will decrease the throughput of the system linearly provided firing time of the SETUP transition is less than that of TT1.

An increase in firing time of the timed transition SETUP will decrease the throughput of the system linearly provided the firing time of timed transition TT1 is less than that of SETUP

An increase in firing time of the transition TT2 has negligible effect on the throughput of the system, but it will delay the production for the very first time, since we are assuming all the jobs are of same kind.

An increase in firing time of the timed transition DELAY has no effect on the throughput of the system, since we are assuming all the jobs are of same kind.

Increasing the firing time of the timed transition PROCESS will decrease the throughput of the system linearly.

Chapter 6

CONCLUSION

The primary purpose of the effort described in this thesis was to define a useful model for a manufacturing workstation. The model was cast as a performance Petri net with color tokens and timed transitions.

Up on examining the result of this thesis it become apparent that initial goal of modeling a manufacturing workstation with the consideration of tool handling, providing raw materials and consumable parts, quality checking and the reuse of reworkable parts has been attained.

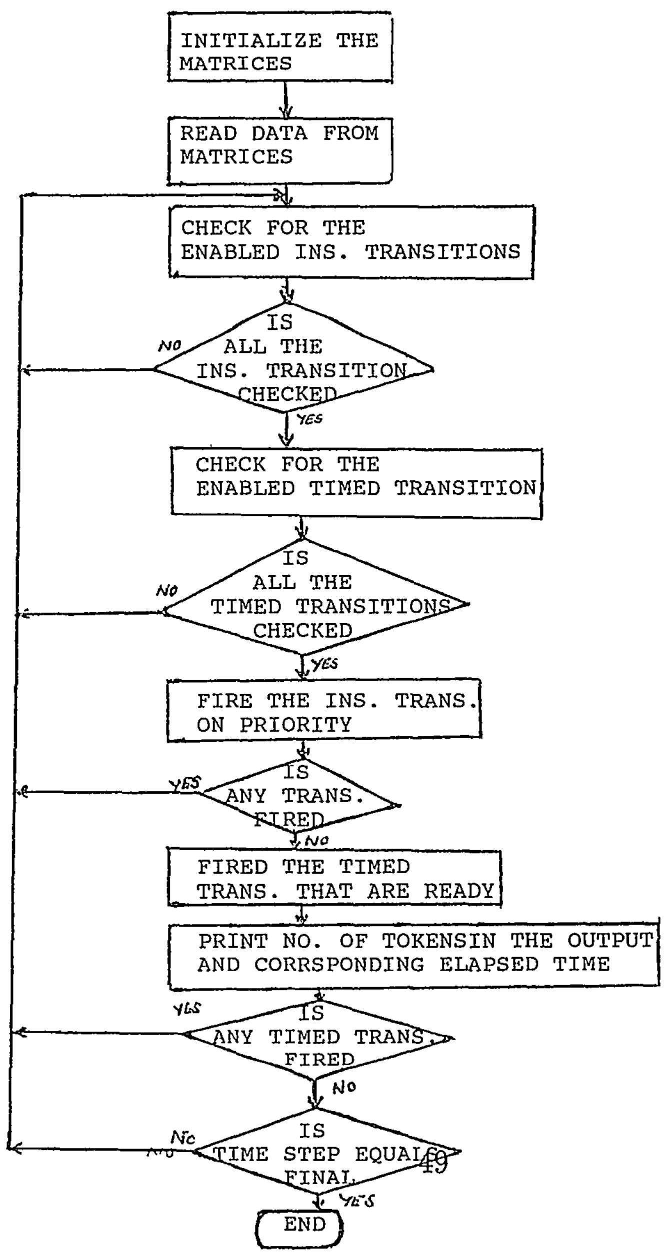
A successful attempt was made to write a simulation program for the simplified Petri net model of the manufacturing workstation by considering all jobs arrived in the workstation are the same.

It has been said that more we learn, the more we learn how little we know. As such, it should come no surprise that this thesis has raised more questions than it has answered. Writing a simulation program for the Petri net model of manufacturing workstation by considering different natured jobs in the workstation and the elimination of the potential dead lock situation in firing the transition from the input place NEXT are recommended for future work.

Appendix A

SIMULATION PROGRAM

A.1 flow Chart for the Simulation Program



A.2 Simulation Program in Basic Language

```
10  REM DEFINE THE TABLE
20  DIM M(17),X(7),Y(5),Z(7),W(5),C(7,17),D(5,17),E(7,17),F(5,17)
30  REM INITIAL MARKING
40  DATA 3,0,0,0,0,0,0,3,0,3,0,0,0,0,0,1,0
50  REM INS. TRANSITION
60  DATA 0,0,0,0,0,0,0
70  REM FIRING OF TIMED TRANSITION
80  DATA 0,0,0,0,0
90  DATA 0,0,0,0,0,0,0
100  DATA 0,0,0,0,0
110  REM INPUT OUTPUT TABLE FOR INS. TRANSITION
120  DATA -1,3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
130  DATA 0,0,-1,1,0,0,0,0,-1,0,-1,0,0,0,0,0,0
140  DATA 0,0,0,0,-1,1,0,0,0,0,0,0,0,0,0,0,0
150  DATA 0,0,0,0,-1,0,1,0,0,0,0,0,0,0,0,0,0
160  DATA 0,-1,0,0,0,0,0,0,0,0,0,-1,1,0,0,0,0
170  DATA 0,0,0,0,0,0,0,0,0,1,0,0,0,-1,-1,0,0
180  DATA 0,0,0,0,0,0,0,0,0,0,1,0,-1,0,-1,0,0
190  REM INPUT OUTPUT TABLE FOR TIMED TRANSITION
200  DATA 0,-1,0,0,0,0,0,-1,1,0,0,0,0,0,0,0,0
210  DATA 0,0,0,0,0,0,0,0,0,0,0,-1,0,1,0,0,0
220  DATA 0,-1,1,0,0,0,0,0,0,0,1,0,0,0,0,0
230  DATA 0,-1,0,0,0,0,0,0,0,-1,1,0,0,0,0,0,0
240  DATA 0,0,0,-1,1,0,0,0,0,0,0,0,0,0,1,0,0
250  REM INPUT TABLE FOR INHEBITED ARC IN INS. TRANSTION
260  DATA 0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0
270  DATA 0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0
280  DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1
290  DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0
300  DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
310  DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
320  DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
330  REM INPUT YABLE FOR INHE. ARC IN TIMED TRANSITION
340  DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
350  DATA 0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0
360  DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
370  DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
380  DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
390   DATA 2,3,3,4,5
400  REM INITIAL MARKING
410   FOR I=1 TO 17
420       READ M(I)
430   NEXT I
440  REM FIRING OF INS. TRANSITION
450   FOR J=1 TO 7
460       READ X(J)
470   NEXT J
480  REM FIRING SEQUENCE OF TIMED TRANSITION
490   FOR K=1 TO 5
500       READ Y(K)
510   NEXT K
520 REM FIRED INS.  TRANSITION
530 FOR J=1 TO 7
540      READ Z(J)
550   NEXT J
560  REM FIRED TIMED TRANSITION
570   FOR K=1 TO 5
580       READ W(K)
590   NEXT K
600  REM INPUT OUTPUT TABLE FOR INS. TRANSITION
```

```
610  FOR J=1 TO  7
620      FOR I=1 TO 17
630          READ C(J,I)
640      NEXT I
650  NEXT J
660 REM INPUT OUTPUT TABLE FOR TIMED TRANSITION
670  FOR K=1 TO 5
680      FOR I=1 TO 17
690          READ D(K,I)
700      NEXT I
710  NEXT K
720 REM INHE. ARC FOR INS. TRANSITION
730  FOR J=1 TO  7
740      FOR I=1 TO 17
750          READ E(J,I)
760      NEXT I
770  NEXT J
780 REM INHE. ARC FOR TIMED TRANSITION
790  FOR K=1 TO 5
800      FOR I=1 TO 17
810          READ F(K,I)
820      NEXT I
830  NEXT K
840   REM FIRING TIME FOR TIMED TRANSITION
850   FOR K=1 TO 5
860       READ T(K)
870   NEXT K
880  FOR R=1 TO 10
890   LET V=0
900             FOR J=1 TO 7
910               LET H=1
920              FOR I=1 TO 17
930        IF C(J,I)=-1    AND M(I)=0   THEN H=0
940         IF E(J,I)=1     AND M(I)>0  THEN H=0
950              NEXT I
960                LET X(J)=H
970            NEXT J
980    FOR K=1 TO 5
990        LET G=1
1000             FOR I=1 TO 17
1010         IF D(K,I)= -1   AND M(I)=0  THEN G=0
1020         IF F(K,I)= 1    AND M(I)>0  THEN G=0
1030             NEXT I
1040         LET Y(K)=G
1050     NEXT K
1060   IF X(5)=1 THEN Y(4)=0
1070   IF X(5)=1 THEN Y(2)=0
1080   IF X(1)=1 THEN Y(2)=0
1090   FOR A=1 TO 7
1100   IF X(A)=1   THEN  Z(A)=1
1110  IF X(A)=1  THEN V=1
1120   LET X(A)=0
1130   FOR I=1 TO 17
1140       FOR J=1 TO 7
1150              M(I)=M(I)+Z(J)*C(J,I)
1160       NEXT J
1170   NEXT I
1180   FOR J=1 TO 7
1190       Z(J)=0
1200  NEXT J
```

Simulation Program

```
1210  NEXT A
1220  IF V=1 THEN GOTO 890
1230  LET L=0
1240  LET U=0
1250  FOR B=1 TO 5
1260  IF Y(B)=1  AND  L<T(B)  THEN L=T(B)
1270  IF Y(B)=1  AND  U=T(B)    THEN GOSUB 1390
1280  NEXT B
1290  LET Q=1
1300  FOR K=1 TO 5
1310     IF Y(K)=1  THEN  Q=0
1320  NEXT K
1330  IF Q=1  THEN P=P+L
1340  IF M(6)>0   THEN GOSUB 1510
1350  LET U=U+1
1360  IF L+1>U   THEN  GOTO 1250
1370  NEXT R
1380  END
1390  LET W(B)=1
1400  LET Y(B)=0
1410  FOR I=1 TO 17
1420     FOR K=1 TO 5
1430        M(I)=M(I)+W(K)*D(K,I)
1440     NEXT K
1450  NEXT I
1460  FOR K=1 TO 5
1470     LET W(K)=0
1480  NEXT K
1490  RETURN
1500  END
1510  PRINT M(6)
1520  PRINT P
1530  PRINT R
1540  RETURN
1550  END
```

Bibliography

[1] P.Alanche, K.Benakour, F.Dolle, P.Gillet, P.Rodrigues and R.Valette *"A Petri Net Based Simulator for Flexible Manufacturing Systems"* in Advances in Petri Nets 1984. LNCS Vol. 188 .Springer Varlay. PP. 1-14 .1985.

[2] Anthony D. Robbi *Manufacturing Workstation Model* Technical Report M:EE 08881 August 1988.

[3] M.Ajmone Marsan, Gianni Conte and G.Balbo *A Class of Generalized Stochastic Petri Net fot the Performance Evalution of Multiprocessor Systems* ACM Trans. on Computer System. Vol.2. No.2. PP 93-122. May 1984.

[4] G.Balbo, G.Chiola, G.Fransceschinis and G.Molinar Roet.*Generalized Stochastic Petri Net for the Performance valution of FMSs* Proc. 1987 IEEE Int. Conf. on Robotics and Automation. PP. 1013-1018.

[5] D.Crockett, A.Desrochers, F.DiCersare and T.Ward. *Implementation of a Petri Net Controller for a Machining Workstation* Proc. 1987 IEEE Int. Conf. on Robotics and Automation. PP.1166-1173.

[6] Dider Dubois and Kathryn E.Stecke *Using Petri Net to Represent Production Process* Proc. 22nd IEEE Conf. on Decision and Control. PP.1062-1067. December 1983.

[7] J.C.Gentina and D.Corbeel *Colored Adaptive Structured Petri Net a Tool for the Automatic Synthesis of Hierarchical Control of FMS* Proc. 1987 IEEE Int. Conf. on Robotics and Automation. PP. 1166-1173.

[8] Giorgio Bruno and Maurizio Marisio *Petri Net Based Simulation of Manufacturing Cells* Proc. 1987 IEEE Int. Conf. on Robotics and Automation. PP. 1174-1179.

[9] Giorgio Bruno and Paolo Biglia *Performance Evaluation and Validation of Tool Handling in FMS Using Petri Nets* Proc. of the Int. Workshop on Timed Petri Nets, Torino, Italy, IEEE Computer Society Press. 1985.

[10] R.Heinonen, A.Jokinen, J.Ranta and B.Wahlstrom *On the Simulation of Flexible Production Systems* Modeling ans Simulation in Engineering. IMACS 1986. PP. 307-310

[11] Hidemi Kodate, Kenichi Fujii and Kuzuyoshi Yamanoi *Rrpresentation of FMS with Petri Net Graph and its Application to Simulation of a System Operation* Robotics and Computer Integrated Manufacturing. Vol. 3. No.3. PP. 275-283. 1987.

[12] James L.Peterson *Petri Nets* ACM Computing Surveys. Vol.9. No.3. PP. 223-252. September 1977.

[13] J.L.Peterson *Petri Net Theory and Modeling of Systems.* Prentice - Hall 1981.

[14] B.H.Krogh and R.S.Sreenivas *Essentially Decision Free Petri Nets for Real Time Resource Allocation* Proc. 1987 IEEE Int. Conf. on Robotics and Automation. PP. 1005-1010.

[15] Manjunatha and N Viswanadham *Application of Petri Net Based Models in the Modeling and Analysis of Flexible Manufacturing System* Proc. of the 1986 IEEE Int. Conf. on Robotics and Automation. PP.312-316.

[16] M.K.Molloy *Descrete Time Stochastic Petri Nets* IEEE Tran. on Software Engineering,Vol. SE 11. No.4. April 1985. PP 417-423.

[17] M.K.Molloy *Performance Analysis Using Stochastic Petri Nets* IEEE Tran. on Computers Vol.C-31. No.9. PP. 913-917. Sept. 1982.

[18] Y.Narahari and N.Viswanadham *Analysis and Synthesis of Flexible Manufacturing System Using Petri Nets* Proc. First ORSA—TIMS Conf. on Flexible Manufacturing Systems, University of Michigan, Ann Arbor. PP. 346-358.August 1984.

[19] Y.Narahari and N.Viswanadham the Invariants of Colored Petri Nets 6th European Workshop on the Application and Theory of Petri Nets. Helsinki. Norway. June 1985.

[20] A.Sahraoui, H.Atabakhche, M.Courvoisier and R.Valette *Joining Petri Nets and Knowledge Based System for the Monitoring Purpose* Proc. 1987 IEEE Int. Conf. on Robotics and Automation. PP.1160-1165.

[21] R.Suri and R.R.Hildebrant *Modeling Flexible Manufacturing System using Mean Value Analysis* Journal of Manufacturing Systems. Vol.3. No.1. 1984.

[22] R.Valette, M.Courvoisier and D.Mayeux *Control of Flexible Production systems* 3rd European Workshop on Application and Theory of Petri Nets, Varenna. Italy. PP.265-277. Sept. 1982

[23] N.Viswanadham and Y.Narahari *Colored Petri Net Models For Automated Manufacturing Systems* Proc. 1987 IEEE Int. Conf. on Robotics and Automation . PP. 1985-1990