Design of Reconfigurable Semiconductor Manufacturing Systems with Maintenance and Failure

Ying Tang and MengChu Zhou
Dept. of Electrical & Computer Engineering
New Jersey Institute of Technology
University Heights, Newark, NJ07102
Email: zhou@njit.edu and yxt8958@njit.edu

Abstract

Due to expensive, highly complex and time-consuming processes, semiconductor-manufacturing systems have been given a special attention. In our previous work [5], a heuristic algorithm to design the reconfigurably automated production system was proposed. However, machine breakdowns and planned and unplanned maintenance were not considered. This paper extends that work and addresses the related design issues in reconfigurable back-end semiconductor manufacturing systems with failures and maintenance. Considering different conditions of machines, queuing network approaches are used to derive the throughput of machines to enhance the proposed virtual production line (VPL) design methodology in which the throughput was originally computed using the deterministic machine processing time. A priority is assigned to each idle machine according to its past performance and adaptive algorithms for reconfiguration are proposed.

1. Introduction

A Reconfigurable Manufacturing System (RMS) is one designed at the outset for rapid change in its structure, as well as its hardware and software components, in order to accommodate rapid adjustment of production capacity and functionality needed in response to new market demands [2]. The National Research Council in a recently released study identified RMS as the number one priority technology in manufacturing for the Year 2020 [9]. Due to the mammoth needs of the semiconductor market, semiconductor manufacturing companies are trying to necessitate the use of RMS, which can be reconfigured and reprogrammed to provide manufacturers with a rapid response capability. Qiu and Wysk introduced the concept of virtual production lines (VPL) to improve the flexibility of the back-end semiconductor manufacturing systems [4]. The fundamental issues for the discrete-event driven VPL design was presented in [5]. However, these methodologies were limited when applied to a system subject to machine failure and periodic maintenance. In the recent academic literature, researchers use Petri nets (PN) to model manufacturing system and catch such dynamics [3, 4]. Queuing theories are also adopted for these analysis and design [1, 7]. Despite these relatively modest activities, it is strongly emphasized that the variability of VPL is a decisive topic to investigate because of the complexity of the semiconductor manufacturing procedures. The primary goal of this paper is to present a method for the adaptive VPL design, which considers the performance of the failure prone machines and periodic maintenance. The rest of the paper is organized as follows. Section 2 considers a VPL with periodical maintenance; Section 3 analyses failure prone machines in a VPL; Section 4 focuses on adaptive algorithms for the VPL design; Section 5 gives an example, and Section 6 is our conclusion.

2. VPL with periodical maintenance

A VPL is organized as a sequence of workcells, each with one or more machines to handle processes in a stage [5]. Machines may have different capacity and efficiency in a workcell. Moreover, idleness of machines due to failure is harmful to the throughput and thus increases the production cost. Therefore, most of machines need to be maintained regularly for a period of time to achieve high availability. In this section, machines’ periodic maintenance is considered first. Some notation for a VPL is defined as follows:

\( i \): machine index
\( j \): workcell index
\( k \): machine class index
Failure periodicity of workcells is have different maintenance time and the maintenance rate of the \(i^{th}\) machine in the \(j^{th}\) workcell of the \(w^{th}\) VPL are calculated as:

\[
\nu_{ijw} = \frac{c_{ijw}}{\tau_{ijw}} \quad (2.1)
\]

\[
\lambda_{ijw} = \frac{\chi_{ijw}}{\sigma_{ijw}} \quad (2.2)
\]

Then, the average speed of the \(j^{th}\) workcell is obtained.

\[
a_{jw} = \sum_{i=1}^{\infty} \nu_{ijw} (1 - \nu_{ijw}) \beta_{ijw} \quad (2.3)
\]

3. VPL with failure and maintenance

Even though periodical maintenance is a useful mean to prevent machines from failure, some malfunctions and exceptions are still inevitable, which in turn cause line imbalance and decrease the speed of workcells. In this section, each machine with failure is modeled as a state machine as shown in Fig. 1. Note that the \(d^{th}\) workcell precedes the \(j^{th}\) one. Meanwhile, this paper assumes that machines may break down only when they are busy.

Let \(\pi_{ijw}, \pi_{F}^{ijw}\) and \(\pi_{E}^{ijw}\) denote the probabilities of the idle, busy, and failure conditions of the \(i^{th}\) machine in the \(j^{th}\) workcell of the \(w^{th}\) VPL, respectively.

![State transition model of the \(i^{th}\) machine in the \(j^{th}\) workcell of the \(w^{th}\) VPL](image)

In this paper, it is assumed that machines may have different maintenance time and the maintenance periodicity of workcells is independent. The speed and the mean maintenance rate of the \(i^{th}\) machine in the \(j^{th}\) workcell of the \(w^{th}\) VPL.
Then solving Equations 3.1 and 3.2 leads to $\pi^t_{ijw}$, $\pi^q_{ijw}$ and $\pi^F_{ijw}$ are obtained:

$$\pi^t_{ijw} = \frac{r_{ijw}}{\Delta} \quad \pi^q_{ijw} = \frac{a_{gw}r_{ijw}}{\Delta} \quad \pi^F_{ijw} = \frac{a_{gw}a_{ijw}}{\Delta}$$

where $\Delta = r_{ijw} + a_{gw}a_{ijw} + a_{gw}r_{ijw}$ and $a_{ijw} = p_{th}$ if the $h$th machine in $M_k$ serves as the $i$th machine in the $j$th workcell of the $w$th VPL.

Due to periodic maintenance, the service speed of a machine is decreased from $u_{ijw}$ to $u_{ijw}(1-t_{ijw})$ where $u_{ijw}$ is the maintenance rate. Thus, considering the effects of both failure and periodic maintenance of machines, equation 2.3 becomes:

$$a_{ijw} = \sum_{i=1}^{e_{ijw}} u_{ijw}(1-t_{ijw})\beta_{ijw}$$  \hspace{1cm} (3.3)

4. Adaptive reconfiguration for the VPL

From the above analysis, it is found that failure of machines is a critical factor that affects their performance. Thus, to select a machine with higher speed and lower failure rate is very important. For example, assume that machines 1 and 2 both can serve jobs in a certain stage $j$. $u_{ijw}$ is larger than $u_{ijw}$, but the failure rate of machine 1 is higher than that of machine 2. Then, the system may frequently spend resources to perform the operation associated with machine 1 without success, and then perform with machine 2 with success. This will certainly decrease the system throughput. In the other words, if the system can select machine 2 prior to machine 1, the system throughput may be improved. Introducing priority for machines and integrating them into VPL design should enable system to adapt for the best reconfiguration of VPLs.

Based on the past performance of machines, the values of $\delta_{hk}$ assigned to the $h$th machine in the $M_k$ is decided as follows:

$$\delta_{hk} = e \quad \text{if } u_{ijw}(1-p_{th})$$

(assuming the $h$th machine in $M_k$ is idle and may serve as the $i$th machine in the $j$th workcell of the $w$th VPL). It is clear that the priority of a machine may change depending on the job it serves.

When a new job comes or other events (e.g., breakdown) happen, the system will update the idle machine pools. If the mean failure rate of a machine is higher than a pre-set number $P_w$ (assuming this machine will serve jobs in the $j$th workcell of the $w$th VPL), remove this machine from its corresponding idle machine pool and add it into $R_k$, where machines get immediate repairs.

At initialization, machines all work in good condition. Due to the statistical data from a real time semiconductor manufacturing system, the mean failure rate of a machine is assumed known. Thus, to configure a VPL for a new work-order, the system orders these machines in $l$ in a decreasing order of their priority values according to work-order information. Then, machines are selected from these sorted idle machine pools according to their values from the highest to the lowest. The extended algorithm is as follows:

**Algorithm 1(Adaptive reconfiguration):**

1. Set $X = E$, the set of VPLs with earliness, and assume the $j$th workcell in a VPL requires the $k$th class of machines.
2. Evaluate the information of the new work-order and calculate the desire speed of the $(u+1)$th VPL:

$$a_{u+1} = \frac{N_{u+1} + s_{u+1} - 1}{Due(u+1)}$$  \hspace{1cm} (4.1)

3. Update idle machine pools: if the failure rate of an idle machine in the $l$ is larger than $P_{f(u+1)}$, move it into its corresponding repair machine pool $R_k$; assign priorities to the remained machines in $l$ and order them in decreasing priority values.

4. Evaluate the system status and calculate the maximum speed of each workcell in the $(u+1)$th VPL (assuming the arrival rate of jobs is larger than speed of any machine in a VPL):

   i). First, calculate the maximum speed of workcells with maintenance:

$$a_{j(u+1)} = \sum_{i=1}^{e_{ijw}} u_{ijw}(1-t_{ijw})\beta_{ijw}$$  \hspace{1cm} (4.2)

   ii). Second, get the minimal speed of the $(u+1)$th VPL only considering workcells with maintenance, that is:

$$a_u = Min \{ a_{jw}, j \in [1, s_u] \text{ and the } j \text{th workcell without failure} \}$$

   iii). Then, calculate speeds of workcells with failures and maintenance (Note that $a_u$ is arrival rates of machines in such workcells):

$$a_{jw} = \sum_{i=1}^{e_{ijw}} u_{ijw}(1-t_{ijw})\beta_{ijw}$$  \hspace{1cm} (4.3)

5. Re-get the minimal speed of the $(u+1)$th VPL according to $a_u = Min \{ a_{jw}, j=1, 2, ..., s_u \}$

561
(6) If $a_{j(a+1)} \geq a_{w+1}$, keep this new VPL running with speed $a_{j(a+1)}$. Then, choose the minimum number of machines $(e_{j(a+1)})$ using Eq. 3.3 such that $a_{j(a+1)} \geq a_{w+1}$ and they are the first $e_{j(a+1)}$ machines in $I_k$, then update $\Phi_k = \Phi_k - e_{j(a+1)}$, and calculate $\Psi(t)$. If $\Psi(t) > 0$, add this new VPL into set $E$ and go to Step (10).

(7) If $X \neq X_0$ do:
   a. Select the $w^{th}$ VPL in the set $X$, remove it from set $X$, and adjust its $a_w$ as follows:
   $$\text{Min } e_{j(w)} \quad j = 1 \text{ to } s_w$$
   Subject to:
   $$\Psi_w \geq 0$$
   $$a_w = \frac{N_w - n_w(t)}{Due(w) - Date(w)}$$
   (4.4)
   (4.5)
   (4.6)
   b. If Step a) succeeds, update $\epsilon_{j(w)}$. Exclude the redundant machines from this VPL and add/move them into idle machine pool $I_k$;
   c. If $\Psi_w = 0$, remove it from the set $E$;
   d. Return to Step (2).

(8) If there is at least one machine available at each stage of the $(u+1)^{th}$ work-order and
   $$\sum_{w=1}^{u} (\Psi_w - \Phi_w) - \Phi_{u+1} > 0$$, configure this new VPL with a slow rate $a_{w(u+1)}$, put this new line into the set $D$ and go to Step (10).

(9) Reject this new order, consider it later and exit.

(10) Calculate the production time for the $(u+1)^{th}$ work-order $T_{u+1}$ and exit:
   $$T_{u+1} = \frac{(N_{u+1} + s - 1)}{a_{u(u+1)}}$$
   (4.8)

Because of tardiness, malfunctions and exceptions during operation, how to reallocate resources to minimize these negative impacts is extremely important. Considering these, an adjustment algorithm is presented based on our previous work [5]:

**Algorithm 2 (Dynamical adjustment)**

System adjusts VPLs when tardiness, malfunctions or exceptions happen:

1. Repeat the following steps for all VPLs:
   a. Calculate $\xi(t), \phi(t), \Psi(t)$;
   b. If $\Psi(t) > 0$, update $E = E \cup \{w\}$; otherwise if $\phi(t) > 0$, update $D = D \cup \{w\}$;

2. If $D \neq \Phi$ do:
   a. If $E \neq \Phi$ for each VPL in set $E$, Reallocation resources through Equations 4.4-4.7; and update $\Phi_k$ and $\Psi(t)$. If $\Psi(t) = 0$, remove it from the set $E$;
   b. Arrange VPLs in $D$ with an ascending order of earliest due dates. For each VPL in $D$, do:

   i) For $j = 1 \text{ to } s_w$ (assuming the $j^{th}$ workcell in the $w^{th}$ VPL requires the $k^{th}$ class of machines)
      a). If $\phi_j \neq 0$, update $I_k$ using the same way as Step 3 in Algorithm 4.1, then, assign the first $\phi_j$ machines in $I_k$ to the $f^{th}$ workcell to increase its speed.
      b). Otherwise, check other VPLs in the system (i.e., $x^{th}$ VPL and the $y^{th}$ workcell in the $x^{th}$ VPL requires the $k^{th}$ class of machines). If there is machine $i$ such that its utilization is less than 0.5, this machine can be shared by $w^{th}$ VPL.

      II) Calculate $a_2 = \text{Min}\{a_{j(w)} \quad j = 1, 2, ..., s_w\}$
      III) If $|B_2 - a_2| \leq 10^{-2}$, remove this VPL from $D$
      IV) Otherwise, update its tardiness $\phi_j$. If $\phi_j > 0$, this order will be delayed $\phi_j$ time to finish based on the present forecast.

5. An example

To better understand the above concepts and methods, the system throughput and machine utilization are introduced first [5].

**Definition 5.1:** The system throughput $g$ is:
$$g = \frac{\sum_{w=1}^{n} N_w}{\text{Max}\{\text{Com}(w)\}; \text{Min}\{ST(w)\}}$$
(5.1)

where $\text{Com}(w)$ is the completion time of the $w^{th}$ work-order and $ST(w)$ is the actual start time of the $w^{th}$ work-order.

**Definition 5.2:** The machine utilization $f$ is:
$$f = \frac{\text{Actual processing time of a machine}}{\text{Total production time}}$$
(5.2)
Then, to demonstrate our methodology, a simplified back-end semiconductor line is used to run three cases. In the conventional case, the system successively processes work-orders according to their order date and at each time, the line is predefined to manufacture certain type of products. In the second case, the system configures a VPL using the methodology presented in [5]. In the third case, the above algorithms are applied to configure and control a VPL. Through these cases, the system is already running and two new work-orders need to be made right now by the following sequential processes: Entry, saw, D/A, curing, plasma, wire bond, inspection and exit. For the simplicity, it is assumed that except in “Saw” and “Wire bond” workcells, machines are reliable. Workcells are equipped with identical machines with different failure rates and machines are fully dedicated to its corresponding workcell.

For these three cases, this paper considers the following instance: before configuring a VPL for the first new work-order, there are ten machines with a failure rate in $I_5$, which can process jobs in “Wire bond” workcell. Their failure rates are $\{0.004, 0.008, 0.01, 0.01, 0.008, 0.008, 0.008, 0.04, 0.04, 0.04\}$. Based on field engineers’ experience, the $P_{iw}$ depends on the $\sigma_{iw}$. In this example, the baseline for the highest failure rate is assumed 0.004 (1/min). On the fifth day of the first new work-order being processed, three machines in the “Saw” workcell break down. The next day after that, eighteen idle machines are added into $I_5$ since an old work-order was finished. These input data are shown in Tables 1 and 2.

Table 1: The input data for two work-orders

<table>
<thead>
<tr>
<th>Work-order</th>
<th>EST</th>
<th>Due</th>
<th>$N_{iw}(mag.)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1st order</td>
<td>1/1/00</td>
<td>1/16/00</td>
<td>10^3</td>
</tr>
<tr>
<td>2nd order</td>
<td>1/3/00</td>
<td>1/13/00</td>
<td>10^4</td>
</tr>
</tbody>
</table>

Table 3 lists our computation results, which compare the changes of speeds of production lines, system throughput and utilization of “Wire bond” workcell, when two work-orders going through the three cases. It is clear to see that in the last two cases, the system can concurrently handle multiple work-orders and adjust the speed of these VPLs according to their conditions and system capacity. Thus, the systems’ throughputs increase by 26.7%. In the third case, due to the priority introduced, system can monitor machines’ performance and immediately repair machines with a high failure rate, that is $p_{iw} > 0.004$, instead of keeping such machines running until periodic maintenance. Thus, the second work-order was finished in advance by one day, and workcell utilization increases by 10.2% and 4.2% than those in the first and second cases, respectively.

6. Conclusion

This paper addresses the design issues in reconfigurable back-end semiconductor manufacturing systems. A queuing network model is used to analyze a workcell’s throughput due to its failure, unplanned and planned maintenance. Because unexpected breakdowns of machines can degrade significantly the system performance, a priority value is introduced for each idle machine. Adaptive algorithms are developed based on this and allow the system to dynamically select machines and adjust the VPL. From the example, the approach is found to be effective in increasing system throughput and machine utilization. To test the proposed algorithms through more complicated cases is our future work.

Reference:


### Table 2: The input data for workcells

<table>
<thead>
<tr>
<th></th>
<th>Saw</th>
<th>D/A</th>
<th>Cure</th>
<th>Plasma</th>
<th>Wire bond</th>
<th>Inspection</th>
</tr>
</thead>
<tbody>
<tr>
<td>$c_{ijw}$ (mag.)</td>
<td>1</td>
<td>1</td>
<td>24</td>
<td>2</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>$\tau_{ijw}$ (min.)</td>
<td>7</td>
<td>6</td>
<td>120</td>
<td>4</td>
<td>30</td>
<td>2.5</td>
</tr>
<tr>
<td>$\chi_{ijw}$ (hour)</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>$\alpha_i$ (day)</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>7</td>
</tr>
<tr>
<td>$\phi_k$</td>
<td>50</td>
<td>50</td>
<td>50</td>
<td>40</td>
<td>150</td>
<td>30</td>
</tr>
</tbody>
</table>

### Table 3: The computation results

<table>
<thead>
<tr>
<th># of days</th>
<th>conventional case ( $a_{\omega}$, 1/min)</th>
<th>2nd case (previous algorithms [5])</th>
<th>3rd case (extended algorithms)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1st work-order</td>
<td>2nd work-order</td>
<td>1st work-order</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1st work-order</td>
<td>2nd work-order</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1st work-order</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>2nd work-order</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th># of days</th>
<th>g</th>
<th>f</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>5789</td>
<td>90.2% (the 5th workcell)</td>
</tr>
<tr>
<td></td>
<td>7333</td>
<td>95.4% (the 5th workcell)</td>
</tr>
<tr>
<td></td>
<td>7333</td>
<td>99.4% (the 5th workcell)</td>
</tr>
</tbody>
</table>