

## Chapter 5

### A Modified Scheduling Algorithm for The FIP Fieldbus System

As we stated before FIP is one of the fieldbus systems, these systems usually consist of many control loops that communicate and interact with each other via single network channel (i.e. cable). Here we will try to develop a scheduling policy that will guarantee a fair distribution of this medium between these control loops, and at the same time takes into account several important parameters in designing such systems<sup>1</sup>. These parameters are:

- (I) Stability of each control loop that is attached in the network.
- (II) Maximizing the network periodic utilization.
- (III) Elimination of the Communication jitter.
- (IV) Maintaining a good worst-case response time for the aperiodic variables.

This algorithm is applicable on both polling and token passing systems with some modifications.

The technique which is used to schedule any FIP system is based on the famous Cyclic Executive Scheduling Algorithm [FIP 98], in which there are two cycles to be defined, one is Elementary cycle or the microcycle, and the other is Major cycle or the Macrocycle (please refer to chapter two of this thesis).

We will propose a way to calculate both these cycles using a model that we will shortly introduce, and then we will build a scheduling table or the BAT as called in the FIP terminology, using the Rate Monotonic (**RM**) algorithm. After that we will apply this modified algorithm using some numerical examples, and at last we will add comments and conclusion about this algorithm.

---

<sup>1</sup> The DCCS called in some references as "Networked Control Systems"[Zhang 2001], [Walsh 2001], and in some references as "Integrated Communication and Control Systems" [Halevi 88].

## **5.1. What are the Advantages we gain from this Algorithm?**

Although there are many approaches which address the problem of scheduling systems utilizing FIP, non of these algorithms discuss all the parameters we mention earlier, but rather it merely address one or two aspect at most, and neglecting the effect of other aspects (constraints). For example, [Kim 98] introduced two new methods to schedule the FIP systems; one is to reduce the length of the Bus Arbitrator Table in order to use a small amount of memory, and the second is to eliminate the communication jitter which is responsible for many disturbances in the system output (please refer to chapter three of this thesis).

Hong in [Hong 95] also introduced a new method that guaranteed the performance of each control loop while maintaining the stability of each loop, and maximizing the network periodic utilization. What is more, Hong's method also suppressed the communication jitter at the same time of the periodic traffic. This scheme was originally adopted from [Halevi 88] who dealt with the same aspects in a slightly different manner.

Ref. [Vasques 94], and [Tovar 99] both tried to find a suitable method to schedule the FIP periodic variables while having a good (low) worst-case response time for the urgent aperiodic variables. All these proposed methods are good from their point of view, but we have noticed some drawbacks on them, which we commented on in chapter three of this thesis.

In our proposal will try to overcome these imperfections. Also we want to demonstrate that some of parameters we mentioned before may contradict with each other, for instance, maximizing the network periodic utilization, may contradict our attempt to find a reasonable worst case response time of the aperiodic traffic. However, that we may cover all these parameters at the same time, but our main concern is towards the stability of each control loop on the bus.

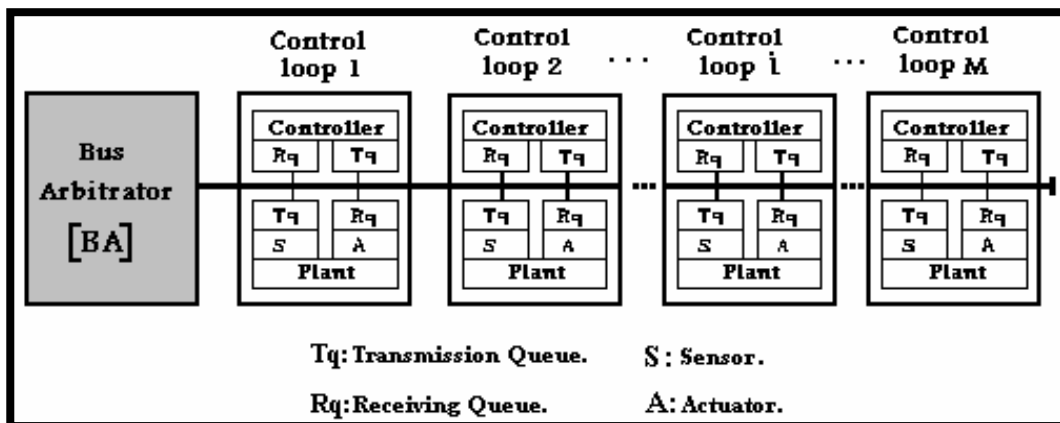
**5.1.2. The Differences between Our Algorithm and the One Proposed by [Hong 95]:**

[Hong 95] mainly dedicated his algorithm to the token passing based systems only, despite he stated clearly more than one time that his algorithm can work on both systems; centralized and token passing. Also he utilized a different approach in choosing the sampling rates of each control loop than the approach we will use. In addition to that, he used a fixed packet size, unlike us. What is more, he maximized the periodic utilization too much beyond the system requirements, without taking into consideration that it may affect the worst-case response time of the aperiodic variables transactions.

**5.2. Identification of the FIP System Model:**

Now we will introduce the model that we will use in this chapter. In **Fig. 5.1.** we can see a typical FIP system, where (**M**) control loops and the Bus Arbitrator (**BA**) all share one network medium (bus). The BA controls the communication process between the nodes along the bus (chapter two).

We assume that each control loop consists of two transmitting nodes; one is the sensor/actuator node, and the other is the digital controller node. Thus the total number of transmitting nodes (**N**) in the medium is giving by: ( $N = 2 * M$ ), in addition to the BA node that has no independent traffic.



**Fig. 5.1.** Typical FIP system.

The sampling rate at the sensor node and at the controller node in any loop  $i$ , is identical and equal to  $T_i$ .

This model is adapted from [Hong 95], which was originally adopted from [Halevi 88] (see Fig. 3.11.), because we want to derive the stability condition of each control loop alone.

Ref. [Halevi 88] stated that a high bit rate is desirable feature in the digital control loops, as it will make the control signals more closely like the continuous signal. But this will mean a heavier network traffic for a given transmission rate, which in turn will increase the data latency of the loops [Halevi 88]. Fig. 5.2 shows the control loop  $i$ , and its components. As shown the control loop system is subjected to two network-induced delays. These delays are:

$\theta_{SCi}$  = delay from the sensor to the controller.

$\theta_{CAi}$  = delay from the controller to the actuator.

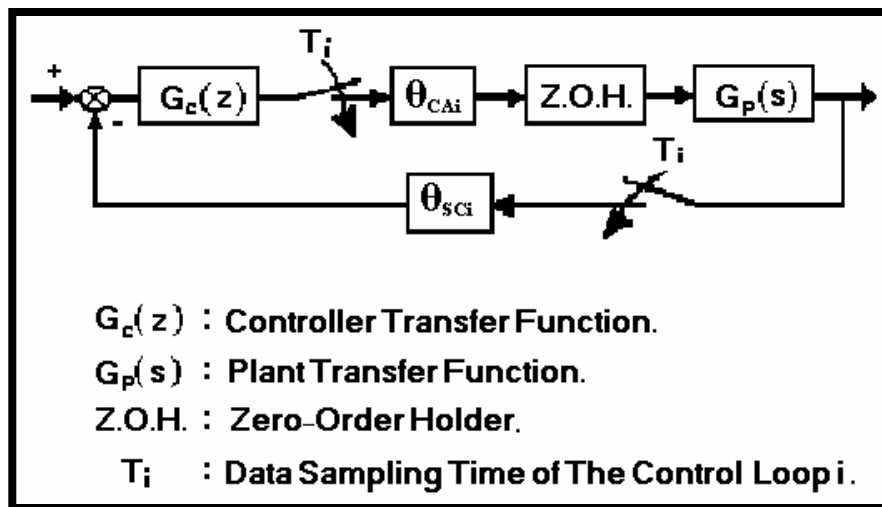


Fig. 5.2. Control loop  $i$  with network delays.

The performance of the feedback control systems could be severely damaged by the time delays introduced by the network [Ray 88]. Since the intervals between polling messages arrival instants at any node are inconsistent (i.e. not synchronized), then the overall network-induced delay ( $D_i$ ) at any node becomes a time varying quantity [Hong 95].

Ray and Halevi showed in [Halevi 88] that the performance of the control loop is directly dependent upon the loop delay. This delay is measured from the instant when the sensor samples a certain data, to the instant the actuator command related to this data, acts upon the actuator (i.e. the command packet arrives at the actuator).

Let us now define the loop delay  $D_i$  of the control loop  $i$ :

$$D_i = \lceil \theta_{SCi} / T_i \rceil * T_i + \theta_{CAi} \quad (5.1.)$$

Fig. 5.3. shows the loop delay  $D_i$  components ( $\theta_{SCi}$ ,  $\theta_{CAi}$ ) of loop  $i$ . From Equation (5.1.), we assumed that  $\theta_{SCi}$  is larger than  $\theta_{CAi}$  and  $T_i$ . This is quite a good assumption, as it is well known that sensors usually do operate on lower frequencies than that of the controllers. So these sensors frequencies may be also lower than the sampling time  $T_i$  of the control loop. One may ask why we did not make the loop delay equals to the algebraic summation of  $\theta_{SCi}$ ,  $\theta_{CAi}$ . This is because both [Halevi 88], [Hong 95] stated that these two delays are not identical in their properties, as we will see later.

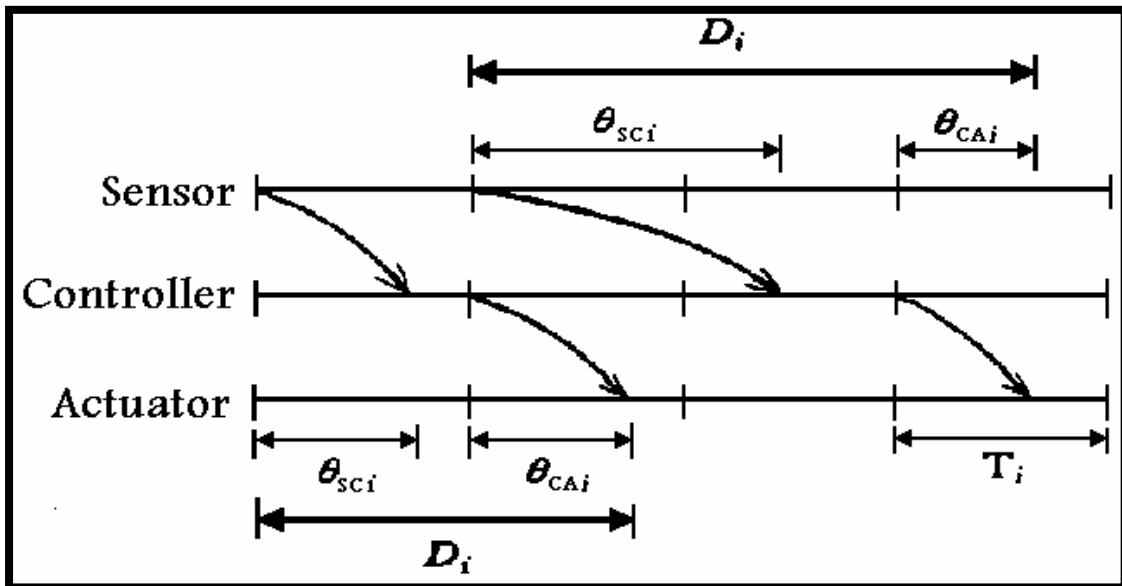


Fig. 5.3. Loop delay of Loop  $i$ .

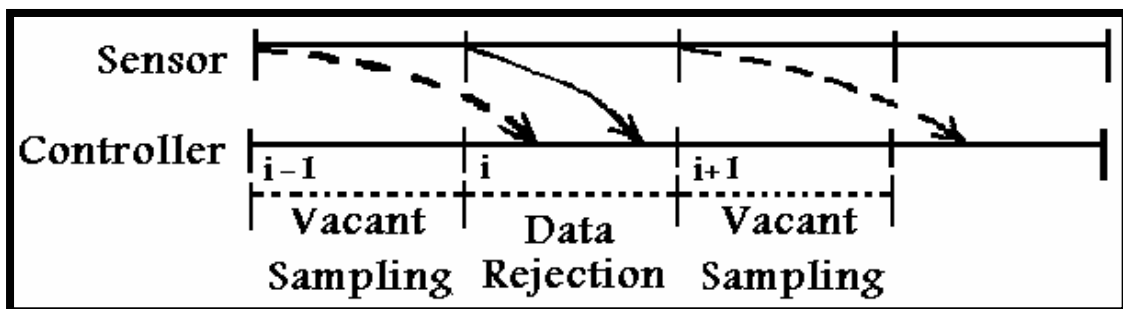
The loop delay  $D_i$  is a time varying quantity, because both  $\theta_{SCi}$  and  $\theta_{CAi}$  are time varying delays.  $\theta_{SCi}$  is time varying and has an upper limit as we will see later.

$\theta_{CAi}$  is also a time varying, that is because even if the controller generates the data packets at fixed time instances (i.e. every  $T_i$ ), but it will encounter delay from the other traffic on the bus, which in turn is a time varying, so these packets will come at inconsistent instances at the actuator.

Unlike the sensor-to-controller data packets which must wait at the receiving queue before being processed by the controller, the controller-to-actuator packets acts directly upon the actuator at the same instance they arrive at their destination (the actuator).

Thus we see that the actuator operates asynchronously, whereas the controller must generate the control packets at a constant rate. From all what we stated before, one can conclude that the characteristics of  $\theta_{SCi}$  are different than that of  $\theta_{CAi}$  [Halevi 88]. In other words, we cannot easily lump those two delays together.

There are two phenomena that happen to the actuator and the controller as a result of this inconsistency of receiving their packets at fixed instances, one is called **Data Rejection**, and the other is called **Vacant Sampling** [Hong 95], [Halevi 88]. See Fig. 5.4 for the physical interpretation of these two phenomena.



**Fig. 5.4.** Example of data rejection and Vacant Sampling.

**Data Rejection:**

When two or more data sample packets come to the controller during the same sampling time instance, then the controller will have to reject the older sample and generate the control signal using only the fresh data sample. This is

can be seen from **Fig. 5.4** at the sampling instance (**i**), the dashed arrow represent the delayed sample from the previous (**i-1**) sampling instance which will be rejected, while the solid arrow represent the new fresh sample which will be used by the controller.

### **Vacant Sampling:**

At some sampling intervals of the controller, no sensor data will come to the controller. So the controller will generate the control signal using the previous sample of the sensor. This can be seen in **Fig. 5.4.**, at either (**i+1**), or (**i-1**) sampling instances as no data reaches the controller.

These two factors may degrade the system performance severely. They also may lead to distortion in the controller signal output, causing undesirable high frequency noise at the actuator input signal. The author of [Hong 95] called it "Excessive Wear".

Our purpose now is to find the sampling times  $T_i$  's of the corresponding control loops, that will satisfy the stability constraints of these control loops, and at the same time increase the network periodic utilization, may be more, to eliminate the communication jitter of periodic variables. Then we will see if this will /or/ will not affect the response time of the aperiodic urgent variables. First of all, in order to deal with feedback control system with variable time delay stability problem, we will use the result found in [Hirai 80]. This result was also adopted by both [Halevi 88], and [Hong 95].

### 5.3. The Feedback Control System Model with Variable Time Delay:

Here we will investigate the stability of a first order system with variable time delay. This system can be modeled by this Equation:

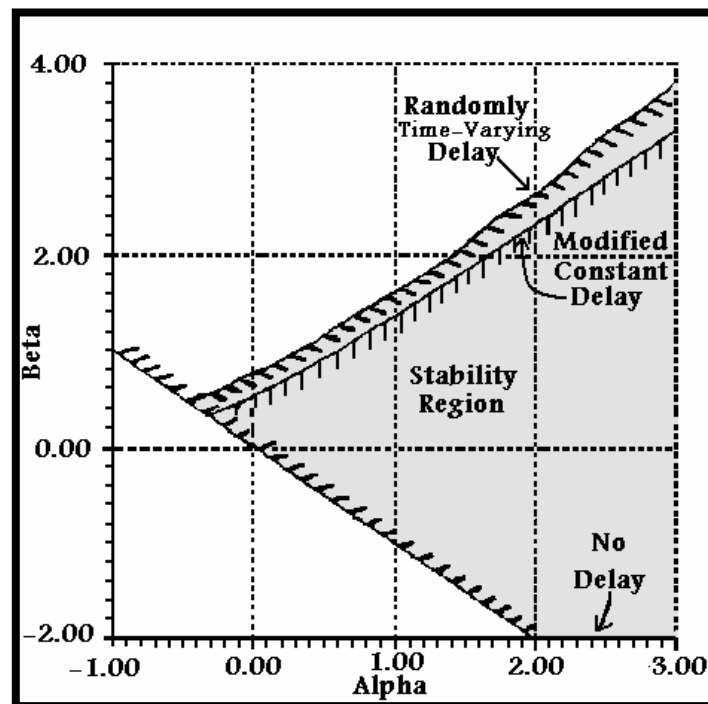
$$\dot{X}(t) + \alpha X(t) + \beta X(t - L(t)) = 0 \quad (5.2.)$$

Where:

$L(t)$  = variable (random) time delay.

$\alpha, \beta$  = Real Constants.

In [Hirai 80], it was stated that if the frozen-time (constant-delayed) system is stable, the delayed system is not always stable unless some conditions are satisfied. The authors sketched the stability margins of (5.2.) in the  $(\alpha - \beta)$  plane for two cases, the variable time delay, and the constant variable delay. This can be seen in Fig. 5.5



**Fig. 5.5.** Stability region for time varying delayed system vs. constant delayed system.

We shaded the stability region with a light gray. The stability region is

bounded by three curves; one is the randomly variable-time delay curve; the second curve is approximately a straight line between 1.0 on the Beta axis and 2.0 on the Alpha axis. This curve represents the values of Beta that makes the system stable. The third curve lies entirely on the Alpha axis and represents the no delay case (i.e. Beta = 0).

From **Fig. 5.5**, one can see that if we replace the randomly variable-time delay  $[L(t)]$ , with a modified constant delay that its value equals to the maximum value of the randomly variable-time delay, then the system will be laid in the stability region. [Hirai 80] had proved that this modified-constant delay is the maximum limit of the delay function  $[L(t)]$ . We would like to say that this conclusion was adopted by both [Halevi 88], and [Hong 95].

### **5.3.1. Modified Loop Delay:**

Define:

$$\Psi_i = \text{maximum allowable loop delay of loop } i.$$

This quantity is derived from the system requirements whether was it a time domain (e.g. Settling Time, Over Shoot, Rise Time ...), or frequency domain (e.g. Phase Margin, Gain Margin) requirements.

In order to find the sampling periods that will satisfy the system stability, we have two conditions that have to be met:

- C1-** Eliminate the rejection and vacant sampling phenomena.
- C2-** Any Loop delay should not exceed its limitation.

In order to meet the first condition, any loop delay component, or in other words  $\theta_{SCi}$  or  $\theta_{CAi}$ , of each loop must be less than or equal to  $T_i$ . In addition, for the fulfillment of the second condition, the over all loop delay must be equal to or less than  $\Psi_i$ .

As we stated before, the loop delay  $D_i$  that we got from Equation (5.1.) is a time-varying quantity; this is mainly because  $\theta_{CAi}$  is a randomly time-varying delay.

The time-variable loop delay is considered a problem. So, if we lump  $\theta_{SCi}$ , and  $\theta_{CAi}$  together, the problem of the overall time-varying delay will still that same. Both [Halevi 88], and [Hirai 80] showed that if we replaced  $\theta_{CAi}$  by its maximum value and substitute in Equation (5.1.) the problem would not be solved yet [Hong 95].

This can be best understood as the effect of time varying delay on the feedback control loop is exerted by the time interval between the two successive actuator commands arrivals [Hong 95]. Since the delay is time-variable, then the interval between two successive actuator commands arrivals is also a time-varying interval. This interval in turn becomes maximum when a minimum delay  $\theta_{CAi}$  at some  $k^{th}$  sampling instance, is followed by maximum delay  $\theta_{CAi}$  at the  $(k^{th} + 1)$  sampling instance.

Now, if we modify the random time-varying delay  $\theta_{CAi}$  to become a constant delay which is the duration of that interval. In other words  $\theta_{CAi}$  to become equal to  $(T_i + \max(\theta_{CAi}) - \min(\theta_{CAi}))$ . Thus we can achieve the stability of the system as stated by [Hirai 80].

Now if we set the maximum of  $\theta_{SCi}$  to become  $T_i$ , as condition (C2) stated before, then we can now define the modified loop delay  $D'_i$  :-

$$D'_i = 2 T_i + [\max (\theta_{CAi}) - \min (\theta_{CAi})] \quad (5.3.)$$

From [Hirai 80], if we replace the random time-varying delay  $[L(t)]$  by  $D'_i$  into Equation (5.2.) it is quit possible to achieve the control system stability [Hirai 80]. Thus, we will use the modified loop delay  $D'_i$  instead of  $D_i$  through out the rest of this chapter to find the suitable sampling time value  $T_i$  that compensates the delay components in any closed-loop  $i$  as we will see later.

## 5.4. The Modified Scheduling Algorithm To Determine The Sampling Times of the FIP Fieldbus.

In this section we aim to find the data sampling of each control loop in the FIP ICCS. In other words, we are trying here to find the periodicity of each periodic variable pair (two variables in one control loop). It's very important to mention that this algorithm is based on the algorithm presented in [Hong 95] with some modification in order to meet the FIP protocol specifications.

### 5.4.1. The Model Parameters:

$\mathbf{T}$  is the vector of sampling times of  $\mathbf{M}$  feedback control system. We can define this vector in mathematical form as:

$$\mathbf{T} = [\mathbf{T}_1, \mathbf{T}_2, \dots, \mathbf{T}_M], \mathbf{T}_i \leq \mathbf{T}_{i+1}, \forall i \quad (5.4.)$$

$L_p$  = length of polling packets in bits.

$L_i$  = length of responding packet of node  $i$  in bits.

$\sigma_1$  = maximum turnaround time in seconds.

$\sigma_2$  = maximum processing time in seconds.

$\mathbf{bps}$  = network speed in bits per seconds.

$L_i = [L_i / \mathbf{bps}]$  = length of responding packets in seconds.

$L_p = [L_p / \mathbf{bps}]$  = length of polling packets in seconds.

$r_i$  = maximum number of data windows that can be served during the worst case latency in loop  $i$ .

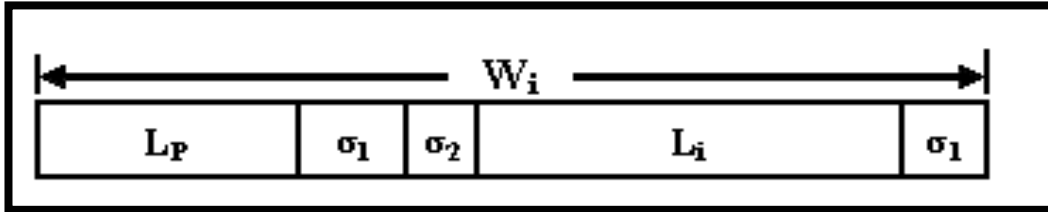
### 5.4.2. The determination of the smallest sampling time $T_i$ :-

Define:

$W_i$  = periodic window duration, which represents one periodic variable transaction.

$$W_i = L_P + L_i + 2*\sigma_1 + \sigma_2 \quad (5.5.)$$

See Fig. 5.6 which demonstrates such window.



**Fig. 5.6.** The duration periodic window slot in FIP system.

The maximum value of loop delay must not exceed its sampling time, as were stated in condition (C1), in order to eliminate data rejection and vacant sampling phenomena. So either  $\max(\theta_{SC})$ , or  $\max(\theta_{CA})$  must equal to  $T_i$ .

For loop 1:

$$\max(\theta_{SC1}) = \max(\theta_{CA1}) = T_1 \quad (5.6.)$$

also, the minimum value of  $\theta_{SC1}$  is:

$$\min(\theta_{SC1}) = L_P + L_i + \sigma_1 + \sigma_2 = W_1 - \sigma_1 \quad (5.7.)$$

Substitute from (5.6.), and (5.7.) into (5.3.) for loop 1:

$$D'_1 = 2*T_1 + [T_1 - (W_1 - \sigma_1)] = 3*T_1 - W_1 + \sigma_1 \quad (5.8.)$$

And from condition (C2):

$$\Psi_1 = 3*T_1 - W_1 + \sigma_1 \quad (5.9.)$$

From Equation (5.9.) we can evaluate  $T_1$ :

$$T_1 = (\Psi_1 + W_1 - \sigma_1) / 3 \quad (5.10.)$$

Let  $r_1$  be the maximum number of periodic variables that can be transmitted during maximum latency in loop 1, so the lower bound on  $r_1$  can be obtained from:

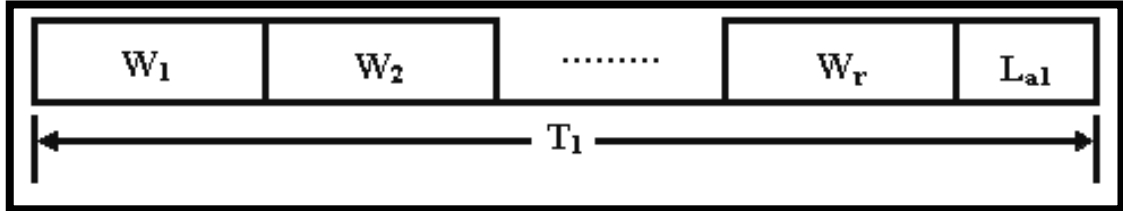
$$\max(\theta_{SC1}) \text{ or } \max(\theta_{CA1}) = r_1 \cdot \max_{i=1,\dots,N}(W_i), \quad (5.11.)$$

$$T_1 = r_1 * \max(W_i) \quad (5.12.)$$

We can rewrite Equation (5.12.) to become:

$$r_1 = r = \lfloor T_1 / \max(W_i) \rfloor \quad (5.13.)$$

From Equation (5.13.), we notice that there is a small amount of  $T_1$  that will remain unused by the periodic data. This amount can be used by aperiodic traffic. Also, it's implied from Equation (5.13.) that the  $T_1$  consists of  $r$  typical windows, this is depicted in Fig. 5.7.



**Fig. 5.7.** The offered time windows in  $T_1$ .

Now, we will generalize Equations (5.6.), (5.7.) to become:

$$\max(\theta_{SCi}) = \max(\theta_{CAi}) = T_i \quad (5.14.)$$

$$\min(\theta_{SCi}) = L_P + L_i + \sigma_1 + \sigma_2 = W_i - \sigma_1 \quad (5.15.)$$

Substitute from both Equations (5.14.), and (5.15.) into condition (C2):

$$\Psi_i = 3 * T_i - W_i + \sigma_1 \quad (5.16.)$$

Rewriting Equation (5.16.) to evaluate the value of  $T_i$  to be:

$$\mathbf{T}_i = (\Psi_i + \mathbf{W}_i - \sigma_1) / \mathbf{3} \quad (5.17.)$$

Equation (5.17.) is a general solution in finding the sampling times  $T_i$ 's, but we must notice that each loop of the  $\mathbf{M}$  loops consists of two transmitting/receiving nodes. From that,  $r_i$  must be even, beside we must make each  $T_i$ ,  $i = 2, \dots, M$ , an even multiple of  $T_1$ , in other words, each  $T_i$  equal  $T_1$  multiplied by some constant, this constant is a power of two. Doing so, we will eliminate the communication jitter that the periodic variables can accommodate [Kim 98].

Define:

$\mathbf{K}$  = Vector of ratios of sampling times with respect to  $T_1$  the smallest sampling time.

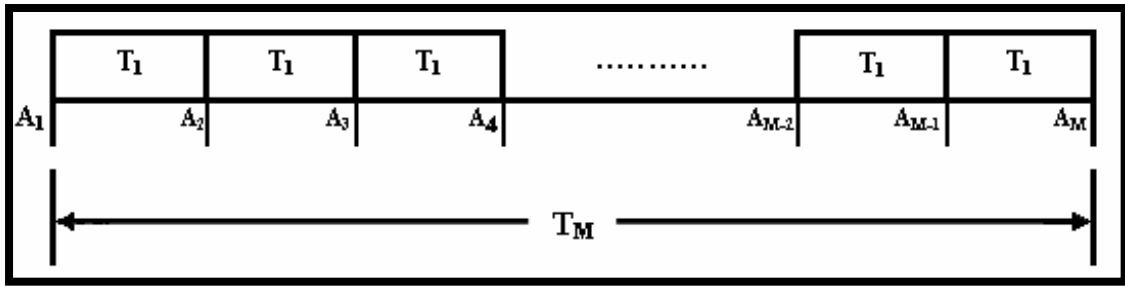
$$\mathbf{K} = [\mathbf{k}_1, \mathbf{k}_2, \dots, \mathbf{k}_M], \quad \mathbf{k}_i = \mathbf{T}_i / \mathbf{T}_1, \quad \mathbf{k}_i < \mathbf{k}_{i+1} \quad (5.18.)$$

Define:

$\alpha_K$  = average number of sampled data during  $T_1$ .

$$\alpha_K = 2 \sum_{i=1}^M (1 / k_i) \quad (5.19.)$$

The verification of Equation (5.19.) can be found in [Hong 95]. If ( $\alpha_k \leq r_1$ ), then all the data sampled during any interval of  $T_M$  will be served by those windows offered by  $T_M$ . **Fig. 5.8.** shows that  $T_M$  consists of a number of equal and identical widows each of  $T_1$  duration.



**Fig. 5.8.**  $T_M$  and its relation to  $T_1$ .

If  $\alpha_K > r_1$ , then the network is overloaded and the algorithm can not accomplish its task, so either we use a higher bit rate network, or to reduce the number of transmitting nodes  $N$ .

Now we will introduce the formula that is used to calculate the rest of  $T_i$ 's sampling time for  $i = 2, 3, \dots, M$ .

Define:

$$T_i = K_i * T_1$$

$$K_i = \langle (\Psi_1 + W_i - \sigma_1) / 3 * T_1 \rangle \quad , i = 2, 3, \dots, M \quad (5.20.)$$

Where  $Y = \langle X \rangle$ , implies that  $Y$  is the power of two (i.e.  $2^{ni}$ ,  $ni \in \Gamma^+$ ), which is closest to  $X$ , but never exceeds  $X$ . If  $(N \leq r)$ , then Equation (5.20.) will equal to Equation (5.17.).

## **5.5. The Construction of the FIP Modified BAT.**

After obtaining the vector of the sampling times, we assign each element in this vector a priority. The highest priority sampling time will be the smallest sampling time (i.e.  $T_1$ ). Then we used the Rate Monotonic (**RM**) algorithm used in [Liu 73] to build the BAT as follows:

- 1) Assign the elementary cycle (microcycle) of the FIP to equal  $T_1$  ( $\mu\text{cy} = T_1$ ).
- 2) Assign the Macrocycle of the FIP to equal to  $T_M$  ( $\text{Mcy} = T_M$ ).
- 3) Define  $t_j$  = sampling instant of loop  $j$  in  $T_M$ .
- 4) Define  $U^k(A_1)$  = number of sampled data from node  $1$  up to  $k$  at  $(A_1)$ , where  $(A_1)$  = beginning instant of the ( $l^{\text{th}}$ )  $T_1$  window in  $T_M$  (see **Fig. 5.7.**).
- 5) Assign  $t_1 = A_1 = 0$ .
- 6)  $t_j = \min [ A_1 \geq A_{1-1} : U^j (A_1) \leq r ], \quad j = 2,3, \dots, N \quad (5.21.)$

This scheduling is said to be optimum if the network utilization for the periodic variables becomes maximum without violating conditions (C1), or (C2). The scheduling optimality of this algorithm is accomplished if and only if  $\alpha_K$  tends to  $r_1$ , [Hong 95].

## **5.6. The Periodic Variables Network Utilization.**

Network utilization is by definition the fraction of time during which the network remains busy transmitting useful periodic data. We have here two mathematical representations for the utilization; the first comes from [Hong 95], and the second come from the definition of the FIP timing constraints. We will present both of them now.

### **5.6.1. The First Mathematical Representation.**

$$U = 2 \sum_{j=1}^M \frac{(L_P + L_j)}{T_j} = \frac{2}{T_1} \sum_{j=1}^M \frac{(L_P + L_j) * T_1}{T_j} \quad (5.22.)$$

This definition which is given by Equation (5.23.), is quite different from the one which was given by [Hong 95] as in this one here uses a variable message size. Equation (5.22.) can be further modified into Equation (5.23.) as follow:

$$U = \frac{2}{T_1} \sum_{j=1}^M \frac{(L_P + L_j)}{k_j} \quad (5.23.)$$

From Equation (5.23.), one can see that if we choose smaller values of  $k_i$ 's then the utilization will increase. Also, this can be done if we decrease  $T_i$ 's, or increase  $M$  as well [Hong 95]. But as we stated earlier, the high increase in the network utilization will lead to increase the response time of the aperiodic variables [Kim 98], [Almeida 2002].

### **5.6.2 The Second Mathematical Representation:**

From chapter four of this thesis we define the binary two dimensional array  $BAT [x] [y]$  to be:

$$BAT[x] [y] = \begin{cases} 1 & \text{if periodic variable (y) is scanned in microcycle (x).} \\ 0 & \text{elsewhere.} \end{cases}$$

$$U = \left( \frac{1}{k_M * \mu c y} \right) \sum_{i=1}^{k_M} \sum_{j=1}^N \{BAT[i][j] * (L_P + L_j)\} \quad (5.24.)$$

Equation (5.24.) is more like the FIP terminology than that Equation(5.23.), as we will verify in the next section of the numerical examples. In addition, it represents the actual periodic traffic.

### **5.6.3. Numerical Example E.5.1.**

Consider one FIP system consists of 5 control loops. Thus the total number of the transmitting nodes equal to 11 ( $2*5 +$  the BA), but as we mention before we will consider the number to be 10.

Also, consider the following information:

$$L_P + L_i = 2 \text{ ms.}$$

$$\sigma_1 = 0.05 \text{ ms.}$$

$$\sigma_2 = 0.0 \text{ ms (i.e. neglect the computation overhead).}$$

$$\Psi_i = \{30, 70, 125, 250, 500\} \text{ ms.}$$

Then we will get the sampling time of the first loop  $T_1$ :

$$W_i = L_P + L_i + 2\sigma_1 + \sigma_2 = 2.1 \text{ ms.}$$

$$T_1 = 10 \text{ ms, } r_1 = 4$$

Calculating the Coefficients  $k_i$ 's:

$$k_1 = 1, k_2 = 2, k_3 = 4, k_4 = 8, k_5 = 16.$$

Check system consistency:

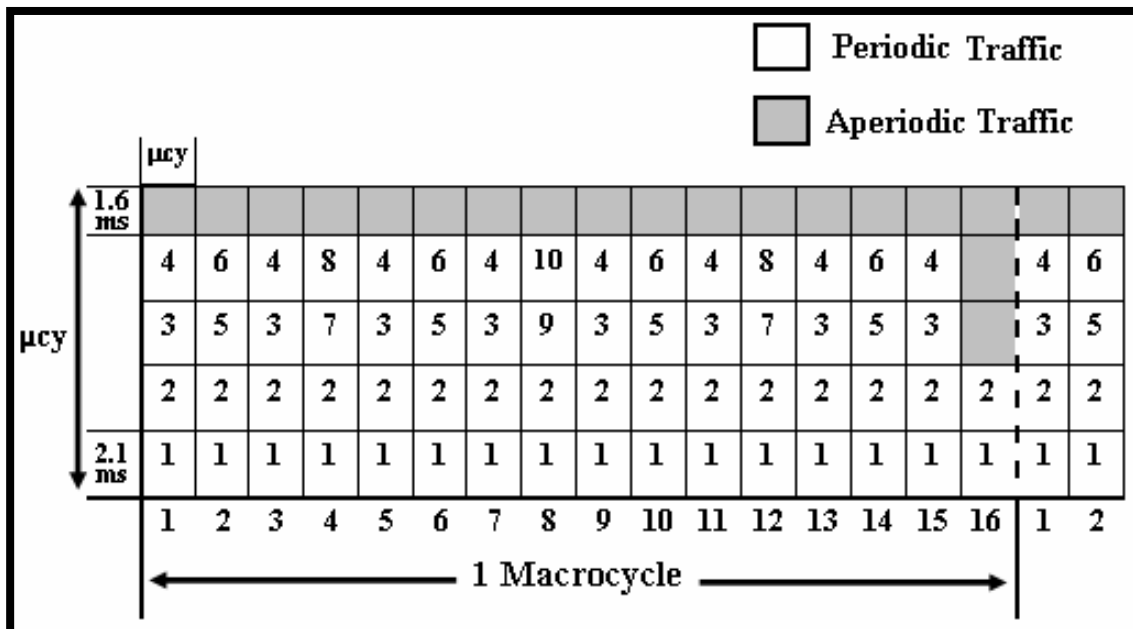
$$\alpha_k = 2 * \{1/1 + 1/2 + 1/4 + 1/8 + 1/16\} = 3.875 < (4 = r_1).$$

Then the system is consistent and we can proceed and construct the BAT after getting the rest of the sampling periods.

$$T_1 = 10 \text{ ms, } T_2 = 20 \text{ ms, } T_3 = 40 \text{ ms, } T_4 = 80 \text{ ms, } T_5 = 160 \text{ ms.}$$

$$\text{Let now } \mu_{cy} = 10 \text{ ms, } M_{cy} = 160 \text{ ms.}$$

The BAT construction can be in **Fig. 5.9**. which shows the BAT we construct from Example E.5.1.



**Fig. 5.9.** The BAT of Example E.5.1.

Calculating the utilization using both definitions:

$$U = 77.5 \%$$

We have some comments that we would like to state about the results of Example E5.1. From those two results, and **Fig. 5.9**, we can see that the remaining time from the microcycles is barely enough to the aperiodic traffic. This is because we make the periodic utilization very high. This will automatically raise the aperiodic response time and its worst case. This can be assured by assuming that the aperiodic transaction window ( $\mathbf{W}_a$ ) to be 2.1 ms. Now if we run those results in our simulation program, we will get a fantastically high aperiodic worst case response time.

One can solve this problem by either increasing the network speed, or by reducing the length of the packets of data. However, this is not always possible. So we will propose a further modification to that algorithm in order to accomplish a good and fair response time for the aperiodic traffic.

## **5.7. A Further Modification to the Modified Scheduling Algorithm.**

We had stated before that  $\Psi_i$ , the maximum allowable time delay of loop  $i$ , is derived from general system requirements. Those requirements can be either time domain, like overshoot, rise time, or settling time, or frequency domain, like phase or gain margins. This indicates that  $\Psi_i$  itself does not involve the stability of the system, but rather the required performance metric(s). So this  $\Psi_i$  has got some range and not an exact value.

Define:

$$\Psi_i \in \{x : \Psi_{i\min} \leq x \leq \Psi_{i\max}\}, x \in \mathcal{R}^+ \quad (5.25.)$$

Without loss of generality, assume  $\Psi_{i\min} = 0.0$ .

$\Psi_{\text{imax}}$  is by definition the upper limit of  $\Psi_i$  beyond it the system will become unstable and its performance becomes unpredictable.

Many researchers tried to find a similar quantity of  $\Psi_{\text{imax}}$ , but instead they choose other metrics rather than the stability [Ryu 95]. We assume that  $\Psi_{\text{imax}}$  is obtained from the control system specifications. We will leave the calculation of  $\Psi_{\text{imax}}$  for future work. We now will use one conclusion that was verified by other resources in order to try to create more time for the aperiodic traffic.

In [Shin 92], [Zhang 2001], and many other as well, it was stated that a delay time interval in the control loop, which is less than or equal to one sample interval of the system, can be tolerated by the control loop and will not affect its stability. This means that, if we delayed the system packets deliberately for an interval less than the system sampling period, the system is still stable. So both this fact and the fact that each  $\Psi_i$  has an allowable range can be used to modify the sampling time intervals of the control loops on the FIP ICCS. We assign the microcycle to be equal to the smallest sampling time.

Define:

$T_1$  = old microcycle and is given by Equation (5.11.).

$W_a$  = maximum time taken to transmit one aperiodic variable.

$(W_a < T_1)$  is a necessary condition to move further in this algorithm.

$$T_1' = T_1 + W_a \quad (5.26.)$$

$$T_i' = k_i * T_1', \quad i = 2, 3, \dots, M. \quad (5.27.)$$

Where  $k_i$ 's is given by Equation (5.20.). From Equation (5.27.), we can see more:

$$T_i' = k_i * T_1' = T_i + (k_i * W_a) \quad (5.28.)$$

From Equations (5.26.), (5.28.), we can see that each variable (or control loop) will encounter a slight constant time delay that is equal to  $(k_i * W_a) \forall i$ . If  $(k_i * W_a < T_i)$  then this delay will be less than one interval.

### **5.7.1. Summary of the Modified Algorithm:**

- A. We apply the algorithm we proposed first to get the sampling times of the control loops.
- B. After we design the system, we test it to see whether it permit us to transmit aperiodic variables or not. If it allows us to do so then no problem and we stop at this step. If not we move on to the next step.
- C. If the system does not contain enough time for the aperiodic transaction, we apply Equations (5.26.), (5.27.) to modify the sampling times of the control loops.

### **5.7.2. Numerical Example E.5.2.:**

Consider the same Example E.5.1., but we will modify the sampling times using the previous modified algorithm to find enough time to the non-periodic traffic. We will get the modified sampling time of the first loop  $T_1$ :

$$W_i = L_p + L_i + 2\sigma_1 + \sigma_2 = 2.1 \text{ ms.}, \text{ assume that } W_a = 2 \text{ ms.}$$

$$T_1 = 10 \text{ ms}, \quad r_1 = 4$$

$$T_1' = T_1 + W_a = 10 + 2 = 12 \text{ ms}$$

Calculating the Coefficients  $k_i$ 's:

$$k_1 = 1, k_2 = 2, k_3 = 4, k_4 = 8, k_5 = 16.$$

Check system consistency:

$$\alpha_k = 2 * \{1/1 + 1/2 + 1/4 + 1/8 + 1/16\} = 3.875 < (4 = r_1).$$

Now we calculate the sampling time values of other control loops of this FIP ICCS system:

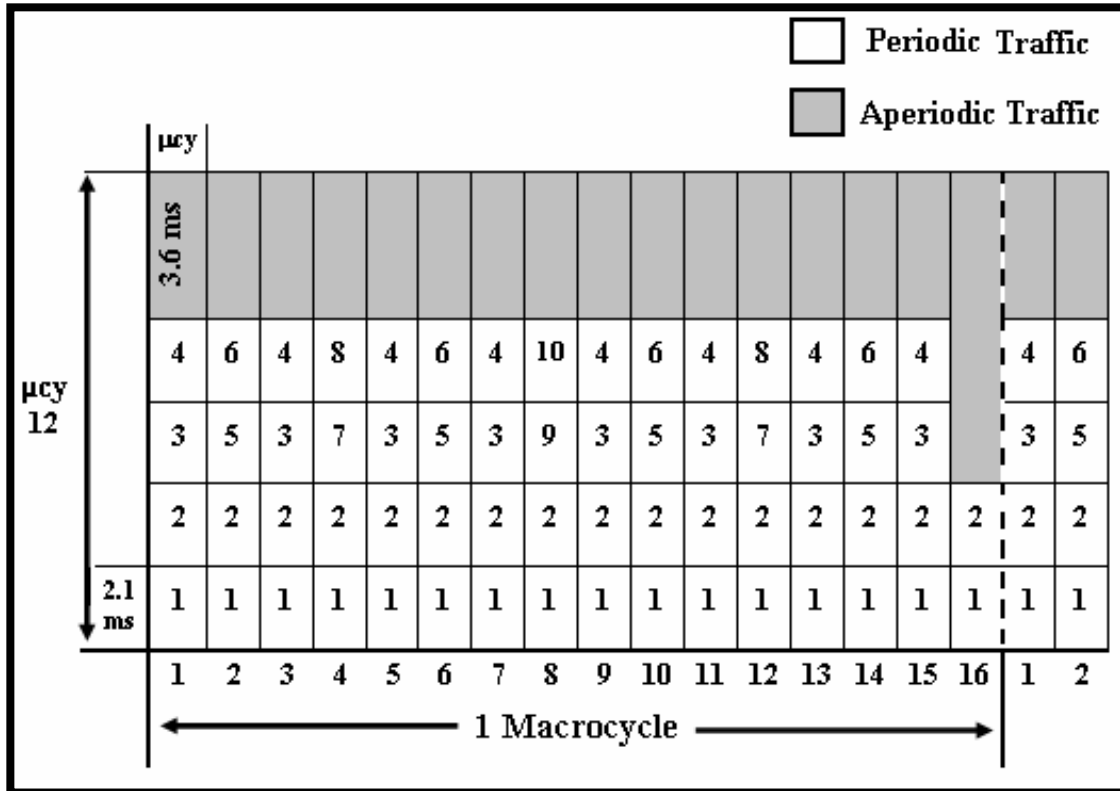
$$T_1' = 12 \text{ ms}, T_2' = 24 \text{ ms}, T_3' = 48 \text{ ms}, T_4' = 96 \text{ ms}, T_5' = 192 \text{ ms}.$$

Let now assign the microcycle and the macrocycle values to be:

$$\mu_{cy} = 12 \text{ ms}$$

$$M_{cy} = 196 \text{ ms.}$$

The BAT construction can be seen in **Fig. 5.10.** which shows the BAT that we constructed from Example E.5.2. but with the new microcycle value.



**Fig. 5.10.** The BAT of Example E.5.2.

We now can calculate the Utilization Using Equations (5.23.), (5.24.) the utilization equals:

$$U = 65.0\%.$$

We see that this utilization in Example E.5.2. is less than that was given in Example E.5.1.; in other words the time that is available for the aperiodic traffic is larger than that of the case of Example E.5.1.

## 5.8. Matlab Simulation.

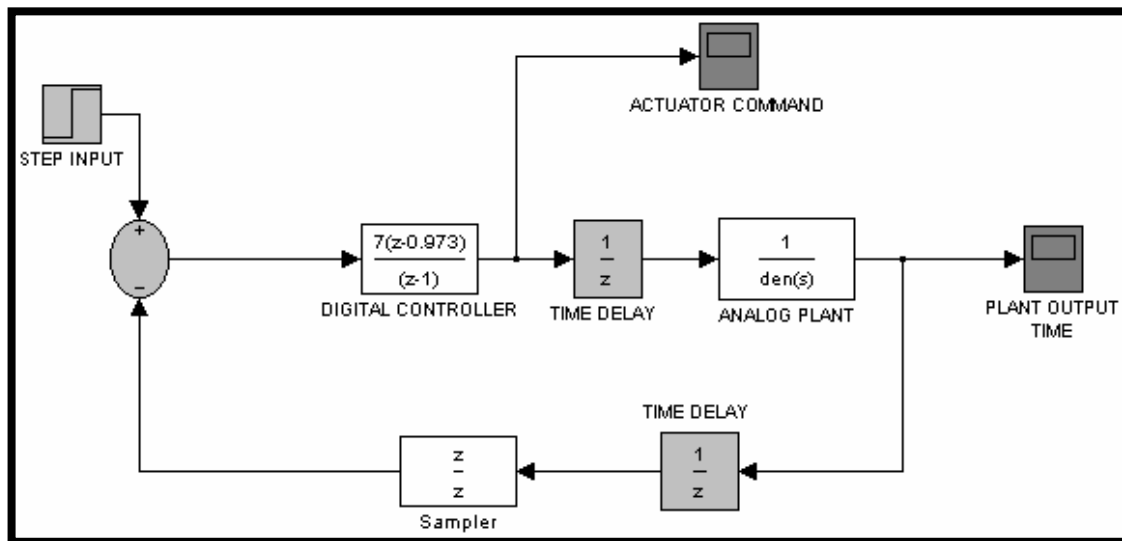
We have constructed a Matlab Simulink<sup>2</sup> model to verify the stability of the system of the closed loop, which was adapted in [Hong 95]. **Fig. 5.11.** shows the model we developed for such loop.

The plant transfer function of loop 1 is:

$$G_P (S) = 1/ (0.009 S^2 + 0.33S + 1) \quad (5.29.)$$

And the analog equivalence of the transfer function of the controller of the loop 1 is:

$$G_C (S) = 7(S + 4)/ S \quad (5.30.)$$

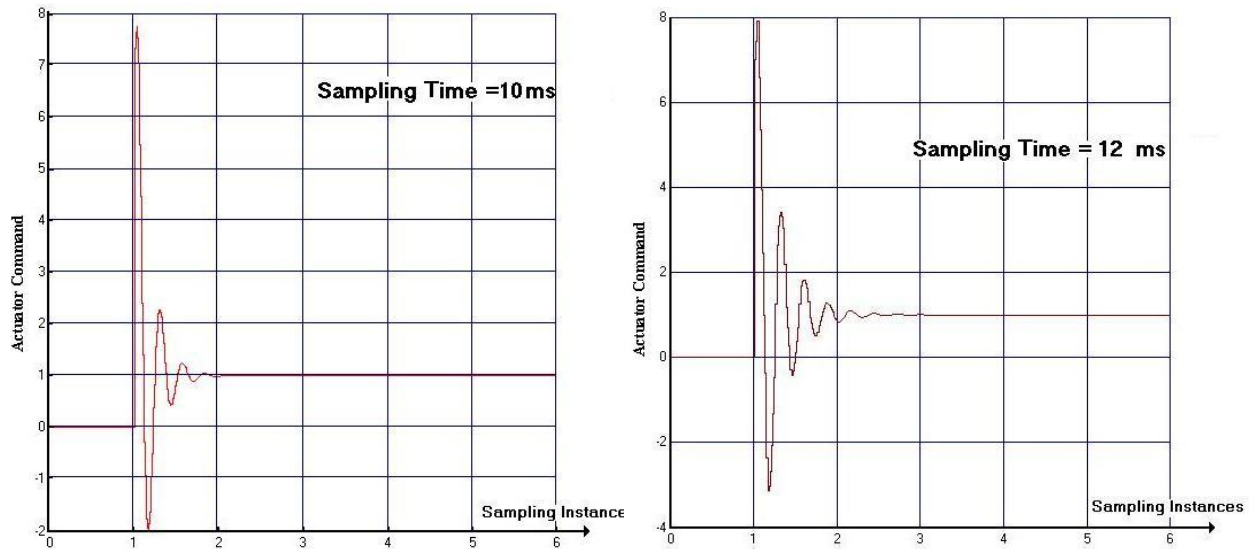


**Fig. 5.11.** The Closed Loop 1 Model.

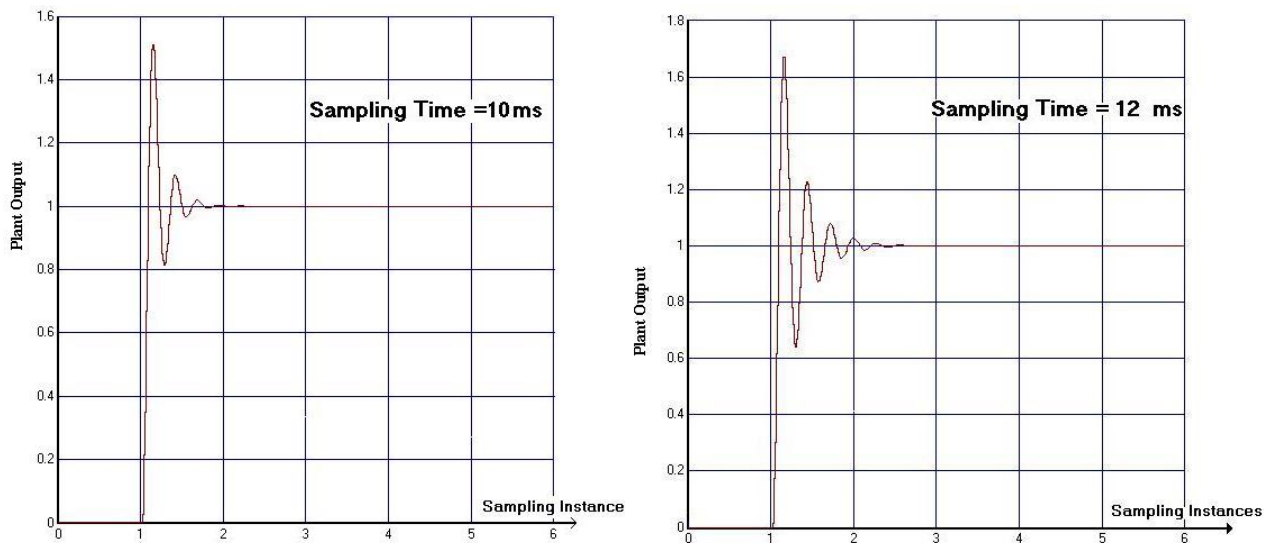
We added a sampler at the feedback branch. As the Simulink does not introduce an independent sampler block, we added a digital transfer function with a zero and a pole at the origin, with unity gain. In addition we would like to add that it is the Simulink only who can mix in simulation between the analog and digital control systems.

<sup>2</sup> We used Matlab® 6.5 Release 13, with Simulink® 5.0.

You can see from **Figures 5.12, 5.13** the difference between the system performances of the first loop for the two sampling times 10, 12 ms. **Fig 5.12.** shows the actuator input, where **Fig. 5.13.** shows the output of the loop.



**Fig. 5.12.** Actuator input of loop 1.



**Fig. 5.13.** Plant output of loop 1.

We can see that the ripples have increased using the new sampling time. However, the system is still stable and the system-overshoot had not increased very much. In addition to that the settling time also did not change.

## **5.9. Conclusion.**

Unlike the algorithm that was given by [Hong 95]; which considered that all the traffic packets are of fixed length, we here modify this by making the packets of variable length. What is more, we have managed to separate the processing time that is taken by each node to process on the polling message ( $\sigma_2$ ) that comes from the **BA**, and the propagation delay of the of the packets itself ( $\sigma_1$ ). These two quantities were often summed up in one value that called “**Overhead**”.

Although the author of [Hong 95] stated clearly in his paper that his algorithm can be applied directly to both token passing systems, and polling systems, we discovered that it's not suitable for the polling systems. For example, he put the general maximum delay  $D_i$  of the loop ( $i$ ), to be  $(N * \sigma)$  which is the total number of stations multiplied in the overhead. This is true only in case of token passing systems only, but it's not true for the case of polling systems like the FIP.

Moreover, he had ignored the token packet transmission time, as if he had lumped it with the data transmission time. Here we also have split both the polling packet from the data packet itself.

We can also add that he assigned a general maximum of  $\theta_{SCi}$ ,  $\theta_{CAi}$  to be  $T_1$  for every node which contradicted his own assumptions that he stated in [Hong 95]. We corrected this situation.

We have introduced a new definition of periodic utilization. In this new definition we utilize the FIP terminology.

One can note that the author [Hong 95] did not take into his consideration the non-periodic traffic and its great importance in the real-time systems. For this particular problem we have proposed another new modified algorithm to deal with it. In this modification we tried to reduce the worst-case response time of the aperiodic traffic by finding slots for it in the BAT.