Simpler Analysis of Serial Transactions Using Reverse Transactions

Karim TRAORE
{karim.traore@ensma.fr}

Emmanuel GROLLEAU
{grolleau@ensma.fr}

Francis Cottet
{cottet@ensma.fr}

LISI/ENSMA
Laboratoire d'Informatique Scientifique et Industrielle
École Nationale de Mécanique et d'Aérotechnique
Téléport 2 – BP 40109 F-86961 Chasseneuil Futuroscope Cedex, France

Abstract[1]

We present in this article a method of validation for "serial transaction". The serial transaction model has been proposed in order to validate a concrete real-time application. This model is typically a task reading serial information (RS232, CAN,...): several instances are identical and read an unitary part of a serial packet, these tasks have the same WCET, offset shifting, priority and relative deadline. In addition, the last task of a transaction has to deal with the whole packet, and is typically longer, but has a longer relative deadline, and a lower priority. The method of validation that we present here uses the concept of reverse transaction deduced from the serial transaction to analyse.

1. Introduction

The model of "serial transaction" has been extracted from a real-time application that consists in developing a mini-UAV (Unmanned Air Vehicle) (figure1). In the development of a real-time application like this one, two techniques of scheduling can be used : the on-line scheduling, with a fixed [8][9][1] or variable allocation of priorities of the tasks in the tasks set [2][7][3] and off-line techniques which use a sequence whose correctness was proved [16][5]. One of the most important phases is the temporal validation which consists in proving that whatever happens, all the tasks meet their temporal constraints. RTA (Response Time analysis) methods are used to bound the worst-case response time of the tasks of an application. Tindell [13] proposed a method for calculating an upper bound of the worst-case response time less pessimistic than classic RTA (considering a critical instant consisting of a simultaneous release of all the tasks) in the context of tasks with offsets (transaction).

Palencia and Harbour [12] extended Tindell's work. Lastly, [15][11] introduced the concept of "imposed" interference differing from "released for execution" interference used by Tindell. However, for now the exact calculation methods used to determinate the exact worst-case response time relies on calculating every combination of the tasks of the transactions; it thus remains exponential in time.

In order to validate the control system of the UAV, we have to deal with tasks with offset which are particular instances of transactions. First, we present the UAV application in section 2. Then section 3 presents the concept of transactions and serial transactions. In Section 4, the method of "imposed interference" is defined. In section 5, we present some new results obtained using the concept of reverse transaction.

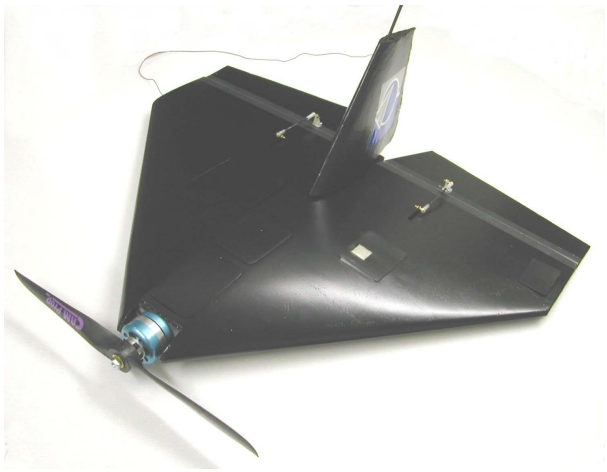


Figure 1: the AMADO

2. Presentation of the Application

The project, named AMADO, is a UAV with a wingspread of 55 cm, using a delta shaped wing with two symmetrical drifts for a total weight (including the control system) of 930 grams. The main objective is to create an autonomous plane embedding a camera, and to be able to follow dynamically defined waypoints. The UAV is connected to a ground station thanks to a wireless modem, allowing it to receive high level

[1] This work was supported by ONERA/DGA

orders during a mission. The critical parts of the flight control are embedded.

2.1 Description of the application

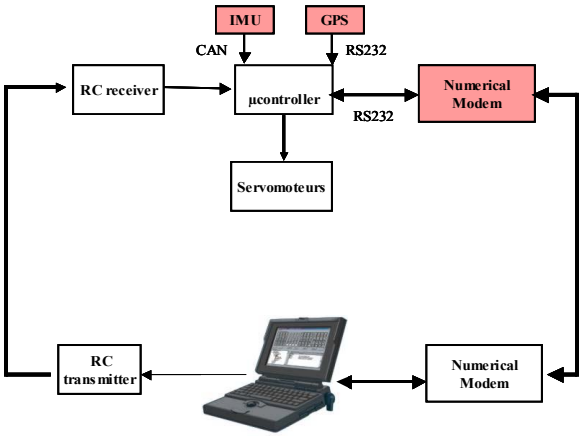


Figure 2: main architecture of the AMADO

The Figure 2 shows two parts: the ground station, and the embedded system.

The embedded system heart is a Freescale/Motorola MPC555[10] connected to the actuators (3 servo-commands and the speed-variator, refreshed every 20 ms), an IMU [6] (Inertial Measurement Unit), a GPS receiver [4], a traditional radio receiver and a modem. The MPC555 is a 32 bits PowerPC with a frequency of 40MHZ, 448KB of flash memory and 26KB of RAM.

Two sensors are used in order to calculate the position and attitude of the UAV: the GPS receiver and the IMU. The Inertial Measurement Unit sends information about angular speed and accelerations, which, once treated, give the roll and the pitch of the UAV. This IMU is connected on a CAN port and delivers information at a frequency of 50Hz and a throughput of 1Mbps. A frame of the IMU is compound of 3 blocks of 6 bytes. In order for the system to get a complete frame, each block must be read before the next arrives. Once the system has 3 blocks, it can constituate the frame, and handle it to calculate the roll and the pitch.

The GPS receiver is used to get the speed (direction and module) and the absolute 3Dimensional position of the UAV. The GPS Receiver sends data to the controller at a frequency of 4Hz and delivers information with a throughput of 57600bps. As a RS232 communication, the information is sent byte after byte; the number of bytes sent during one period (frame) of the GPS can reach 120 bytes. As in the case of the IMU, the system must recover each byte and arrange it before the arrival of the next byte, under penalty of losing the complete frame.

Finally the modem [Modem1] connected to the microcontroller on the serial port is bi-directional and communicates with the microcontroller at a throughput of 115kbps. The length of the frame transmitted to the microcontroller by the modem can reach 10 bytes. The requirements are the same as in the case of the GPS receiver. In the presentation of this architecture, we omitted voluntarily the video circuit that does not have any impact on the real-time aspects of this application.

2.2 Software architecture of the application

We have chosen the real-time executive OSEKTurbo OS/MPC5xx of Metrowerks[19] for our application. This RTOS is conforming to the standard OSEK/VDX [18]; standard defined for applications with limited resources.

Apart the initialisation task, there are 12 tasks in the control system (see Table 1). The priorities of the tasks have been assigned following a Deadline Monotonic policy [8]. Note that the value L=120 (resp. L=3, L=10) corresponds to the number of times the task has to be activated in order to acquire a frame.

Tasks	Period	WCET		deadline	Priority
	(in microsecond)				
Monitoring (1)	200000	60		200000	1
Acq PWM (2)	20000	24		10000	7
Transmit Grd (3)	50000	3360		30000	5
Deliver Cmd (4)	20000	40		10000	6
Navigation (5)	250000	560		140000	2
ReguleAttitude (6)	60000	32400		60000	4
Acq GPS (7)	250000	100	L=120	160	11
Acq IMU (8)	20000	96	L=3	720	10
Acq Instruction(9)	100000	12	L=10	80	12
TreatGPS (10)	250000	3000		5000	9
TreatIMU (11)	20000	900		7500	8
TreatInstruction (12)	100000	900		70000	3

Table1: task system of the UAV

This kind of application cannot be validated easily if the offsets are not taken into account. Indeed, it appears clearly that task TreatGPS is released when the whole GPS frame has been received; it cannot thus be released at the same time as the task Acq GPS; it is the same case for task TreatIMU and the task Acq IMU; the same situation occurs for the task TreatInstruction and the task Acq Instruction.

3. Presentation of serial transaction

The Figure 3 presents a model of a serial transaction, L_i instances of the acquisition of a part of a frame are separated by a duration corresponding to the

arrival rate of the packets (Acq GPS, Acq IMU, Acq Instruction), and a longer task is used to handle the whole frame (TreatGPS, TreatIMU, TreatInstruction). In a serial transaction, the acquisition tasks are usually short, because they only have to bufferize the packets until the whole frame is built, while the treatment tasks are longer since they have to deal with the full frame.

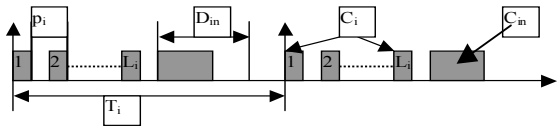


Fig. 3. pattern of a serial transaction

Let us give some results found in [13][14][12] relative to transactions. Certain tasks can have the same period and be bound by relations of offsets i.e. they can never be released at the same time. A set of tasks of the same period bounded by offset is called a transaction. Let us note $\Gamma := \{\Gamma_1, \Gamma_2, \ldots \Gamma_k\}$ a set of transactions. A transaction Γ_i contains $|\Gamma_i|$ tasks with a period T_i :$\Gamma_i := <\{\tau_{i1}, \ldots, \tau_{i|\Gamma i|}\}, T_i>$.[12][14]

A task is defined by $\tau_{ij} := <C_{ij}, O_{ij}, D_{ij}, J_{ij}, B_{ij}, P_{ij}>$ where C_{ij} is the worst-case execution time (WCET), O_{ij} is the offset, D_{ij} is the relative deadline, J_{ij} the maximum jitter, B_{ij} maximum blocking due to lower priority tasks, and P_{ij} the priority. Without loss of generality, we consider that the tasks are ordered by non decreasing offsets O_{ij} ; in our case, we define the response time as being the time between the release of the task and the completion of the task.

Let us note also $hp_i(\tau_{ua})$ the set of indices of the tasks of Γ_i with a priority higher than the priority of a task τ_{ua} i.e. $j \in hp_i(\tau_{ua})$ if and only if $P_{ij} > P_{ua}$.

The task under analysis is usually noted τ_{ua}. Tindell showed that the critical instant of τ_{ua} is a particular instant when it is released at the same time as one task of higher priority in each transaction (its own transaction being handled separately). The main difficulty is to determine what is the critical instant candidate τ_{ic} of a transaction Γ_i that initiates the critical instant of τ_{ua}. An exact calculation method would require to evaluate the response time obtained by carrying out all the possible combinations of the tasks of priority higher than τ_{ua} in each transaction and to choose the task τ_{ic} in each transaction that leads to the worst-case response time. This exhaustive method has an exponential complexity and is intractable for realistic task systems; several approximation methods exist.

The first method of approximation is the "released for execution" method. This method has been very pessimistic in the case of the application of mini UAV; it has been enable to validate this application however we have proved that the application is valid. Up to now, the best known method of approximation is the "imposed interference" method.

4. Upper bound method based on the "imposed" interference

This method has been proposed in [15]; it removes the unnecessary overestimation taken into account in the classic computation of the interference imposed by a task τ_i on a lower priority task τ_{ua}. This overestimation does not have any impact in the case of tasks without offset but has a considerable effect in the approximation of the worst-case response time when we are in the presence of tasks with offsets.. This method consists in calculating the interference effectively imposed by a task τ_j on a task τ_{ua} with a lower priority during a time interval of length t. In order to calculate this “imposed” interference, [15] subtracts a parameter x (see Figure 2) from the original interference formula; let us note $W_{ic}(\tau_{ua}, t)$ the interference of Γ_i on the response time of τ_{ua} during a time interval of length t when τ_{ic} is released at the same instant as τ_{ua}.

$$W_{ic}(\tau_{ua}, t) = \sum_{j \in hp_i(\tau_{ua})} \left(\left(\left\lfloor \frac{t^*}{T_i} \right\rfloor + 1 \right) * C_{ij} - x_{ijc}(t) \right)$$

$$t^* = t - phase(\tau_{ij}, \tau_{ic}) \qquad phase(\tau_{ij}, \tau_{ic}) = (O_{ij} - O_{ic}) \% T_i$$

$$x_{ijc}(t) = \begin{cases} 0 & \text{for } t^* < 0 \\ \max\left(0, C_{ij} - (t^* \% T_i)\right) & \text{for } t^* \geq 0 \end{cases}$$

Example: this transaction has 4 tasks with period $T_i = 50$

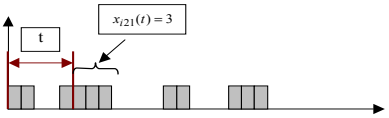


Fig. 2: imposed interference

$\Gamma_i := < \{\tau_{i1}, \tau_{i2}, \tau_{i3}, \tau_{i4}\}, 50 >$

$\tau_{i1} := < 2, 0, 4, 0, 0, 4 >$

$\tau_{i2} := < 4, 4, 8, 0, 0, 2 >$

$\tau_{i3} := < 2, 12, 5, 0, 0, 3 >$

$\tau_{i4} := < 3, 17, 15, 0, 0, 1 >$

$W_{i1}(\tau_{ua}, 5) = (2 - 0) + (4 - 3) + (0 - 0) + (0 - 0) = 3$

For determining the upper bound of the response-time, we use this function :

$$W_i(\tau_{ua}, t) = \max_{c \in hp_i(\tau_{ua})} W_{ic}(\tau_{ua}, t)$$

With the value of each $W_i(\tau_{ua}, t)$, the response time R_{ua} of τ_{ua} can be calculated. $R_{ua}^{(n+1)} = C_{ua} + \sum_{i \in \Gamma} W_i(\tau_{ua}, R_{ua}^n)$. R_{ua} is obtained by fix-

point iteration starting with $R_{ua}^0 = C_{ua}$. Let us execute this method on the example (Figure3 (a))

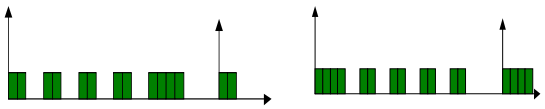


Fig. 3.(a and b) Example for imposed interference (a) reverse transaction (b)

In the transaction Γ_i, we have five tasks. Let us consider a lower priority task τ_{ua} with $C_{ua} = 5$. Let us calculate the response-time. We present at first the details of iteration number 2:

Iteration 2:

$W_{i1}(\tau_{ua},5) = (2-0)+(2-1)+(0-0)+(0-0)+(0-0) = 3$

$W_{i2}(\tau_{ua},5) = (0-0)+(2-0)+(2-1)+(0-0)+(0-0) = 3$

$W_{i3}(\tau_{ua},5) = (0-0)+(0-0)+(2-0)+(2-1)+(0-0) = 3$

$W_{i4}(\tau_{ua},5) = (0-0)+(0-0)+(0-0)+(2-0)+(4-3) = 3$

$W_{i5}(\tau_{ua},5) = (0-0)+(0-0)+(0-0)+(0-0)+(4-0) = 4$

$W_i(\tau_{ua},0) = 4 \qquad R_{ua} = 9$

We give the values obtained in the different iterations in the table below:

Iteration number	t	W_{i1}	W_{i2}	W_{i3}	W_{i4}	W_{i5}	W_i	R_{ua}
1	0	0	0	0	0	0	0	5
2	5	3	3	3	3	4	4	9
3	9	5	5	5	6	5	6	11
4	11	6	6	7	6	6	7	12
5	12	6	6	8	6	6	8	13
6	13	7	7	8	7	7	8	13

Consequently, the value of R_{ua} is equal to 13.

It is not simple to evaluate the value of imposed interference. Indeed, with this method it is necessary in each iteration to evaluate the value of "n" interferences with "n" as the number of tasks in the transaction. Moreover, it is necessary to evaluate the value of " x_{ijc} " "n" times in each iteration. In order to simplify the calculation of the value of imposed interference, we use the transaction presented in figure 3(b). We call this transaction the reverse transaction Γ_i^{-1} of the transaction Γ_i. With the reverse transaction, we show in the next section that it is sufficient to calculate only the value of $W_{i^{-1}1}(\tau_{ua},t)$ at each iteration.

We present in the table below the values obtained in the different iterations:

Iteration Number	1	2	3	4	5	6
$W_{i^{-1}1}$	0	4	6	7	8	8

5- Reverse transaction method

Definition1: A serial transaction is a transaction with the following constraints:

Let Γ_i be a serial transaction,

- null jitter: $\forall i/\tau_{ij} \in \Gamma_i$, $J_{ij}=0$
- regular arrival pattern p_i: $\forall j \in [1..|\Gamma_i|]$, $O_{ij}=(j-1)p_i$.
- there are two kinds of tasks :
  - the $L_i=|\Gamma_i|-1$ acquisition tasks such that : $\tau_{ij,j\in[1..Li]} := <C_i, (j-1)p_i, p_i, 0, B_{ij}, P_i>$;
  - the treatment task $\tau_{i|\Gamma i|}:=<C_{in},L_ip_i,D_{in}, 0, B_{ij}, P_{in}>$
- with $C_{in}>C_i$, $D_{in}>p_i$,$P_{in}<P_i$ and $(T_i - L_i \cdot p_i) - C_{in} > p_i - C_{in}$. This means that the treatment task is longer than the acquisition tasks, but is provided a longer deadline and a lower priority.

Definition2 : a task τ_{ua} is an intermediate priority task for a serial transaction Γ_i if the priority of τ_{ua} is lower than acquisition tasks of Γ_i but higher than the treatment task of Γ_i.

Definition3 : a task τ_{ua} is a lower priority task for a serial transaction Γ_i if the priority of τ_{ua} is lower than all the tasks of Γ_i.

Definition4: Let Γ_i be a serial transaction, we call reverse transaction of the serial transaction Γ_i the transaction Γ_i^{-1} obtained by putting in first position the task of last position of Γ_i; the other parameters remain identical (period, offsets between tasks, etc.) (see figure 3 (a and b)). The tasks of Γ_i^{-1} are defined as:

- $\tau_{i^{-1}1} = <C_{in}, 0, C_{in} + (p_i - C_i), 0, B_{i|\Gamma_i|}, P_{i|\Gamma_i|}>$
- $\tau_{i^{-1}j,j\in[2..|\Gamma_i|]} = <C_i, C_{in} - C_i + (j-1)\cdot p_i, p_i, 0, B_{i(j-1)}, P_{i(j-1)}>$

Theorem1: Let Γ_i be a serial transaction, let Γ_i^{-1} be its reverse transaction and τ_{ua} a task under analysis. If τ_{ua} is a lower priority task for the serial transaction Γ_i, then the interference imposed by the serial transaction Γ_i on the task τ_{ua} when it is released at the same time as the task initiating the critical instant in Γ_i has exactly the same value as the interference imposed by Γ_i^{-1} on τ_{ua} when τ_{ua} is released at the same time as the first task in transaction Γ_i^{-1} i.e $W_{i^{-1}1}(\tau_{ua},t) = W_i(\tau_{ua},t)$ for any t.

Proof: : Let us note $f_i = T_i - L_i \cdot p_i$; and $W_{i^{-1}1}(\tau_{ua},t)$ the imposed interference on the task τ_{ua} by the transaction Γ_i^{-1} in a time interval of length t. We thus will calculate $W_{i^{-1}1}(\tau_{ua},t) - W_{ic}(\tau_{ua},t)$.

For any time interval of length t, we know that there is an integer k such that $t = k \cdot T_i + t \,\%\, T_i$.

According to [17], $W_{i^{-1}1}(\tau_{ua},t) = W_{i^{-1}1}(\tau_{ua},k\cdot T_i) + W_{i^{-1}1}(\tau_{ua},t\%T_i)$

$W_{ic}(\tau_{ua},t) = W_{ic}(\tau_{ua},k\cdot T_i) + W_{ic}(\tau_{ua},t\%T_i)$

Since the value of interference imposed in any time interval of length T_i (Period) is the same whatever the beginning of this interval is, then $W_{i^{-1}1}(\tau_{ua},k\cdot T_i) = W_{ic}(\tau_{ua},k\cdot T_i) = k\cdot(L_i\cdot C_i + C_{in})$

consequently,

$W_{i^{-1}1}(\tau_{ua},t) - W_{ic}(\tau_{ua},t) =$

$W_{i^{-1}1}(\tau_{ua},t\%T_i) - W_{ic}(\tau_{ua},t\%T_i)$

so we can suppose $0 \le t < T_i$; with this consideration, we have :

Interval	$W_{i^{-1}1}(\tau_{ua},t)$	$W_i(\tau_{ua},t)$
$t \le C_{in}$	$W_{i^{-1}1}(\tau_{ua},t) = t$	$W_{in}(\tau_{ua},t) = t$
$t \ge C_{in} + L_i\cdot p_i$	$W_{i^{-1}1}(\tau_{ua},t) = C_{in} + L_i\cdot C_i$	$W_{i1}(\tau_{ua},t) = C_{in} + L_i\cdot C_i$
$C_{in} < t$ and $t < C_{in} + L_i\cdot p_i$	$W_{i^{-1}1}(\tau_{ua},t) = C_{in} + \left\lceil \frac{t-(C_{in}+(p_i-C_i))}{p_i} \right\rceil \cdot C_i - x_{i^{-1}1}(t)$	$W_i(\tau_{ua},t) = \max\left\{ W_{ic}(\tau_{ua},t) \;\middle|\; \forall\, c \in [1..L_i+1] \right\}$

For $t \le C_{in}$ and $C_{in} + L_i\cdot p_i \le t < T_i$, we have already $W_{i^{-1}1}(\tau_{ua},t) = W_i(\tau_{ua},t)$

We have now to prove the equality $W_{i^{-1}1}(\tau_{ua},t) = W_i(\tau_{ua},t)$ for $C_{in} < t < C_{in} + L_i\cdot p_i$.

For $t \in]C_{in}; C_{in}+(p_i-C_i)]$, $W_{i^{-1}1}(\tau_{ua},t) = W_{i(L_i+1)}(\tau_{ua},t)$; c is equal to L_i+1 ; and for $t \in]C_{in}+(p_i-C_i); C_{in}+L_i\cdot p_i[$, $\exists c \in [1..L_i]$ such as $t \in]C_{in}+(p_i-C_i)+(L_i-c)\cdot p_i; C_{in}+(p_i-C_i)+(L_i-c+1)\cdot p_i]$

For these two cases, we have $W_{i^{-1}1}(\tau_{ua},t) = W_{ic}(\tau_{ua},t)$

Moreover, $W_{ic}(\tau_{ua},t) = (L_i - c + 1)\cdot C_i + (C_{in} - x_{icn}(t))$ because $f_i - C_{in} > p_i - C_i$ (according to the definition of serial transaction).

We will prove now that for all $p \in [1..L_i+1]$, $W_{ip}(\tau_{ua},t) \le W_{ic}(\tau_{ua},t)$

We take the value of $W_{ic}(\tau_{ua},t)$ like reference (figure 4 (a))

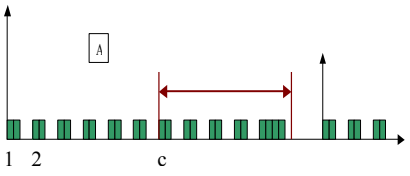
1st case : p > c with c<L_i+1

It appears clearly on the figure 4 (A and B) that the shifting of the interval t from c to the position p decreases the value of the interference by $(p-c)\cdot C_i$ on the left side whereas the increasing on the value of interference obtained on the right side is lower or equal to $(p-c)\cdot C_i$ because $f_i - C_{in} > p_i - C_i$. Therefore $W_{ip}(\tau_{ua},t) \le W_{ic}(\tau_{ua},t)$.

2nd case : p < c with c>1

On the figure 4 (A and C), we can see that every time the interval t is shifted the value p_i towards the left (until we reach the position p such as $t < (L_i+1-p)\cdot p_i$), we add C_i on the value of the interference on the left side; however, the decreasing on the value of the interference on the right side is in the interval $[C_i; p_i]$; therefore the value of the interference after this shifting decreases. When t is lower than $(L_i+1-p)\cdot p_i$, a shifting of the interval t towards the left doesn't change the value of the interference (Figure 4 (D)).

Consequently, the value of $W_{ip}(\tau_{ua},t)$ is always lower or equal to the value of $W_{ic}(\tau_{ua},t)$.



Example of Calculation of $W_{ic}(\tau_{ua},t)$ with c=7

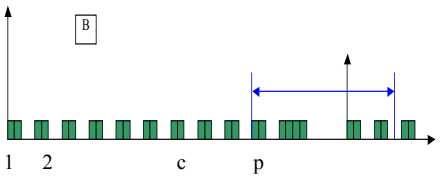
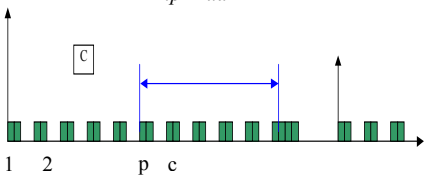
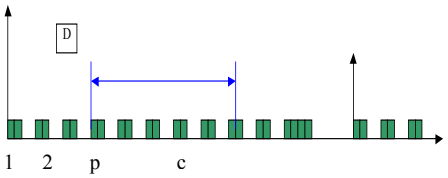


Illustration of $W_{ip}(\tau_{ua},t)$ with p>c



1st Illustration of $W_{ip}(\tau_{ua},t)$ with p<c



2nd Illustration of $W_{ip}(\tau_{ua},t)$ with p<c

Fig.4 : Illustration of the comparison between $W_{ip}(\tau_{ua},t)$ and $W_{ic}(\tau_{ua},t)$

□

Let us note $hp(\tau_{ua})$ the set of serial transactions such that τ_{ua} has a lower priority than every task of the transaction.

By applying Theorem1, the interference applied by the serial transactions whose indices belong to $hp(\tau_{ua})$ in a time interval of length t is :

$$\sum_{j\in hp(\tau_{ua})}\left(\left\lceil\frac{t}{T_j}\right\rceil\cdot C_{jn}+\left\lfloor\frac{t}{T_j}\right\rfloor\cdot L_j\cdot C_j\right.$$

$$\left.+\min\left(\left\lceil\frac{(t\%T_j-C_{jn}-(p_j-C_j))}{p_j}\right\rceil,L_j\right)\cdot C_j\right)$$

This formula facilitates the calculation of the upper bound of worst-case response time for the lower priority tasks and allows us to validate the mini UAV application. Indeed, using this formula in conjunction with the formula obtained in [17] for the intermediate priority tasks, we obtained the following values (Table 2) :

Tasks	Period	deadline	Priority	Worst-case response time
1	200000	200000	1	56156
2	20000	10000	7	6532
3	50000	30000	5	15532
4	20000	10000	6	6572
5	250000	140000	2	56096
6	60000	60000	4	54636
7	250000	160	11	124
8	20000	720	10	468
9	100000	80	12	12
10	250000	5000	9	3408
11	20000	7500	8	5620
12	100000	70000	3	55416

Table2: Calculation of tighter upper bound of worst-case response time of all the tasks of the UAV application

In the table 2, we can see that all the upper bound worst-case response times are lower than the deadline; consequently the application of the UAV is valid.

6– Conclusion

In this article, we have at first presented the model of serial transactions. A serial transaction Γ_i is compound with L_i short but urgent acquisition tasks activated each time a serial packet is received, and a less urgent but longer treatment task activated when a whole frame is received.

The number of acquisition tasks can be important and makes the exact calculation of response time intractable. Then, we have presented the imposed interference method that gives a tighter upper bound .

After these presentations, we have introduced the concept of reverse transaction that simplify the way to evaluate the imposed interference of a transaction. Our future work will focus on the method to determine the real worst case response-time in the context of serial transactions.

References

1. N.C. Audsley, Optimal priority assignment and feasibility of static priority tasks with arbitrary start times, Tech. Report YCS-164, University of York, nov. 1991.
2. M.L. Dertouzos, Control robotics : the procedural control of physical processors,
Proc. of IFIP Congress, 1974, pp. 807-813.
3. M.L. Dertouzos, A.K. Mok, Multiprocessor on-line scheduling of hard real-time tasks, IEEETransactions on Software Engineering 15(12), Déc. 1989, 1497-1506.
4. TIM-LC, TIM-LF, TIM-LP System Integration Manual, http://www.u-blox.com
5. E. Grolleau, Ordonnancement temps réel hors-ligne optimal à l'aide de réseaux de pétri en environnement monoprocesseur et multiprocesseur, thèse, ENSMA - Université de Poitiers, nov. 1999.
6. Crista Inertial Measurement Unit (IMU) Interface / Operation Document, May 2004, http://www.cloudcaptech.com.
7. J. Labetoulle, Un algorithme optimal pour la gestion des processus en temps réel, Revue Française d'Automatique, Informatique et Recherche Opérationnelle (Fév.1974), 11-17.
8. C.L. Liu and J.W. Layland, Scheduling algorithms for mutltiprogramming in real-time environnement, Journal of the ACM 20(1) (1973), 46-61.
9. J. Leung and J. Whitehead, On the complexity of fixed-priority scheduling of periodic, real-time tasks, Performance Evaluation (Netherland) 2(4) (1982), p.237-250.
10. MPC555/MPC556 User's Manual October 2000, http://e-www.motorola.com
11. J. Mäki-Turja and M. Sjödin, Improved Analysis for Real-Time Tasks With Offsets –Advanced Model. Technical Report MRTC no. 101, Mälardalen Real-Time Research Centre(MRTC), May 2003.
12. J.Palencia Gutierrez and M.Gonzalez Harbour. Schedulability Analysis for Tasks with Static and Dynamic Offsets. In Proc. 19th IEEE Real-Time System Symposium (RTSS), December 1998
13. K. Tindell, Addind Time-Offsets to Schedulability Analysis, Technical Report YCS 221, Dept of Computer Science, University of York, England, January 1994
14. J.Mäki-Turja and M.Nolin. Faster Response Time Analysis of Tasks with Offsets. In Proc. 10th IEEE Real-Time Technology and Applications Symposium (RTAS), May 2004
15. J.Mäki-Turja and M.Nolin. Tighter Response Time Analysis of Tasks with Offsets. In Proc. 10th International conference on Real-Time Computing and Applications (RTCSA'04), August 2004
16. J. Xu and D.L. Parnas, Pre-run-time scheduling of processes with exclusion relations on nested or overlapping critical sections, Phoenix Conference on Computers and Communications (Phoenix, USA), Apr. 1992,pp. 6471-6479.
17. K.Traore, E.Grolleau and F.Cottet, Efficient Schedulability Analysis of Serial Transactions, rapport de recherche nr 06001, Laboratoire d'Informatique Scientifique et Industrielle ENSMA, Janvier 2006
18. OSEK/VDX operating system specification 2.2.2 July 2002, http://www.osekvdx.org.
19. OSEKturbo OS/MPC5xx User's Manual, Juin 2003, http://www.metrowerks.com.