

AN OVERVIEW ON CLOCK SYNCHRONIZATION SOLUTIONS

with focus on software algorithms

José Alberto Fonseca, Pedro Fonseca¹

{jaf,pf}@det.ua.pt
Departamento de Electrónica, Universidade de Aveiro
3810 AVEIRO Portugal

Abstract: Clock synchronization allows the establishment of a global time base, a requirement for many distributed systems applications. The properties that can be expected from clocks in a computer system are presented and formalized. Some of these properties are not present in the physical clock, and they must be enforced by means of clock synchronization. Several solutions can be found in the literature. A classification for is presented, which will make easier the choice of a clock synchronization solution for a specific system. *Copyright ©2000 IFAC.*

Keywords: Clock synchronization; distributed systems; algorithms.

INTRODUCTION

Frequently, the correct operation of a real-time distributed system assumes (or requires) the existence of a global time base. Computer clocks, running autonomously, cannot provide this time base. The developer of such a system is required to implement some means of clock synchronization. There are several solutions available, and the difficulty is sometimes to choose the best from a set of alternatives. In this paper, we try to provide some assistance to this, by looking at the different solutions that have been proposed and trying to organize them in a systematic way. We expect this can make easier the choice of a clock synchronization algorithm. In section 1, we present the properties that can be expected from a set of clocks, and in section 2 we describe the actual implementation of a clock. The existing solutions for clock synchronization are classified in section 3.

1. CLOCKS IN DISTRIBUTED SYSTEMS

Computer clocks allow us:

- to recover the sequence of past events by assigning each one a time-stamp;
- to measure the time elapsed between two instants;
- to start one action at a given instant in time, or after some time has been elapsed since some other previous event;
- ...

Programs like ‘make’ reconstruct the sequence of the actions that have been performed (which files have been saved, ...) based on the file’s time stamps. For make to act correctly, we need clock values that increase as time goes by. We will call this property *monotonicity*. When computers time-out some action, we expect time intervals, as measured by the computer, to be close to the actual physical time interval value. In other words, we allow some drift from actual physical time, but we expect it to be bounded. We call this the *bounded drift* property. Another requirement arises when files are to be shared and time-stamped

¹ Corresponding author.

by two computers. For the sequence of actions to be correctly reconstructed from the time marks, clocks in two different computers must show approximately the same value at same time: *precision*. Finally, we may want the computer to start a given action at a given time. For instance, you may want the display to show the message “Time to go home” at 6:00 p.m.. This requires the value displayed by the clock to match the standard time — at the risk of having to explain your boss (or your spouse) why leaving work so soon (respect., so late) (*accuracy*).

These properties can be expressed in a more formal way. $C_p(t)$ represents the clock of computer p at physical time t . Monotonicity means that:

$$\forall t_1, t_2, \quad t_1 > t_2 \Leftrightarrow C_p(t_1) > C_p(t_2) \quad (1)$$

From the property of bounded drift, we need the intervals as measured by the computer clocks to be approximate to the actual time intervals. The quality of the approximation is measured by the *clock drift*, ρ :

$$(1 - \rho) \leq \frac{C_p(t_1) - C_p(t_2)}{t_1 - t_2} \leq (1 + \rho) \quad (2)$$

Precision is formalized by defining the quantity δ , the clock synchronization precision. The clocks of sites p and q are synchronized to a precision δ if and only if:

$$\forall t, \quad |C_p(t) - C_q(t)| \leq \delta \quad (3)$$

For defining accuracy, we need to quantify how close a clock should be to standard time. Being α the accuracy, a clock C_p is accurate to α if and only if:

$$\forall t, \quad |C_p(t) - t| \leq \alpha \quad (4)$$

What we can expect from computer clocks depends on these four properties: monotonicity, bounded-drift, precision and accuracy.

2. THE PHYSICAL CLOCK

Probably all computer clocks in the world are based on the same device, the *physical clock*. The structure of a physical clock is shown in figure 1. It is based on an oscillator and a counter. The oscillator is supposed to produce periodic events, the clock *ticks*, at a certain frequency, the clock’s *nominal frequency*, f_{nom} . At each clock tick, the counter is incremented by a value g , the clock’s *granularity*.

The oscillator’s actual frequency, $f_{\text{osc}}(t)$, is often different from the nominal frequency, f_{nom} . The deviation of the frequency from its nominal value

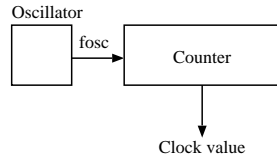


Fig. 1. Physical clock structure.

is caused by several factors. In quartz crystals, these can be the temperature sensitivity, the effect of coupling capacitors and aging. The frequency drift ρ is defined as an upper bound on the absolute value of the fractional deviation of the frequency value with respect to its nominal value. It can be shown that clock drift and frequency drift are equivalent.

Usually, $f_{\text{osc}}(t) \simeq f_{\text{nom}}$, which implies $\rho \ll 1$. Most commonly, the oscillator is based on a quartz crystal; for this type of oscillator, the value of ρ is typically 10^{-6} to 10^{-5} . Imperfections, either in the crystal or in the resonant circuit, can make this value grow up to 10^{-4} , and drifts of 2.5×10^{-4} have been observed (Mills and Thyagarajan, 1994). Quartz oscillators exhibit a good performance at low cost, weight and power; (Frerking, 1978; Vig, 1992) present quartz oscillators in more detail. Other solutions exist with better timing characteristics, such as caesium or rubidium atomic clocks. But these are heavier, more expensive and demand more power, which makes them inadequate for a great deal of applications.

Due to the existence of frequency drift, two clocks that are set to the same time value will eventually differ with no bounds on this difference; the same applies to a clock being set to some standard time.

3. SOLUTIONS FOR CLOCK SYNCHRONIZATION

The physical clock is a device that we assume to possess the properties of monotonicity and bounded drift, but not the properties of precision nor accuracy. If these are required by the application, additional means should be employed to guarantee them. That is the subject of clock synchronization.

Several solutions exist for synchronizing clocks in a computer system. We will now address the problem of classifying them. This classification will make easier the choice of a clock synchronization solution for a specific system. We prefer the word solutions to the most commonly found algorithms, as we want to consider methods where the concept of algorithm may not be easily perceivable (at least, from the user’s point of view).

We can identify three main classes of clock synchronization solutions (Fonseca *et al.*, 1998):

- Broadcast
- Hardware
- Software

This classification is similar to the one proposed in (Olson and Shin, 1994). Notice that only the last two classes are oriented towards distributed systems. In the first, it is irrelevant whether the sites are integrated in a distributed system or not: each site behaves like an autonomous entity in face of the reference site.

Broadcast based clock synchronization Broadcast based clock synchronization uses timing signals broadcast by some site equipped with a highly accurate clock. These sites can be accessed by means of a telephone call, by getting the signals sent by some radio stations or through the Global Positioning System (GPS).

In telephone call clock synchronization, the time provided by an accurate clock is received by the computer's modem, encoded in some text or binary format. This solution requires the installation of a modem and access to the time server to be accessible at a reasonable price. The message format is highly variable, depending on the service provider.

Several radio stations (WWV and WWVH in the USA, DCF77 in Germany, France Inter in France and others) provide timing information on their radio signal. Radio receivers equipped with a decoder are able to extract the timing information from the radio signal. A survey of these methods can be found in (Lichtenecker, 1997).

By using radio signals, the need for dedicated cabling is eliminated. Putting clock synchronization to work based on this type of solutions has some aspects that cannot be neglected. Firstly, its implementation requires using blocks that are inserted in the system as black boxes. Choice will be limited to the solutions that are proposed and prevents development of specific solutions, which could be more adapted to the system.

Sites to be synchronized must be located where radio reception is possible with the required quality. This can prevent usage of this solution inside buildings or in industrial environments.

The receiver interface should be compatible with existing hardware and software. The physical dimensions of the receiver and antenna should be considered. Power consumption can be of concern, namely when we look for autonomous devices. Finally, price is another issue to consider. The receivers' cost can be insignificant for applications like synchronizing computers at a worldwide

banking system or at a power distribution plant. But in the case of low-cost distributed systems, its cost may be prohibitive. In table 1 we summarize the main aspects that affect clock synchronization by means of radio broadcast.

Aspect	Impact
Products developed by a third-party	Limit to the freedom of choice
Signal reception Interface	Site location Compatibility with existing hardware and software
Size	Dimensions of synchronized unit
Power consumption	Autonomy
Price	Cost of system (mult. by number of sites)

Table 1. Radio broadcast based clock synchronization.

The Global Positioning System (GPS) can also be used to synchronize clocks. GPS is based on a constellation of 24 satellites. For GPS to provide a service with full quality, at least 4 satellites should be visible by the receiver. GPS provides the user two types of services: the Standard Positioning Service (SPS) and the Precise Positioning System (PPS).

The Standard Positioning Service, which is the GPS service available to any user, provides a time transfer accuracy with 340 nanoseconds with a 95% probability and an absolute position estimation with an error of 100m (95%) horizontally and 156m (95%) vertically (Dana, 1997). In SPS, the quality of signals available to the ordinary (civilian) user is intentionally degraded. This signal degradation is named SA, for Selective Availability.

The more accurate Precision Positioning System, with no signal degradation, is only available to users authorized by the U.S. military, and allows time transfer with an error of 200ms (95%) and position estimates with errors of 22m (95%, vertical) and 27.7m (95%, horizontal). For more details, the interested reader is addressed to (Dana, 1997). (Wannenmacher and Halang, 1994) and (Sterzbach, 1997) provide some examples of clock synchronization by using GPS signals.

The remarks about radio based synchronization apply to GPS. In addition, the usage of GPS as a time source presents two particularities. The first is the settling time. After power on, the system may take as long as 12 minutes before full QoS is achieved (under the assumption that 4 satellites are in line of sight). Another potential problem with GPS is that it is a military system. It is owned and run by the US Department of Defense and, although the majority of users are civilians,

the decisions concerning GPS are primarily driven by the interests of the US military.

Aspect	Impact
(All aspects of radio-based CS)	
Settling time	Service delay
Military system	Service availability

Table 2. GPS based synchronization.

Hardware clock synchronization In hardware based clock synchronization, the clock signals of one or several sites are distributed to all the sites in the system. Each site generates then a reference signal, based on the incoming clock signals, that is used to drive or correct the local clock oscillator.

This solution provides tight clock synchronization at the expenses of dedicated wiring (to carry the clock signals) and dedicated hardware (generation of the reference signal). The fact that the local clock oscillator is driven by the CS hardware implies that this solution must be considered at the stage of designing the system. It is not easy to integrate hardware based clock synchronization in a existing system, which wasn't designed for this kind of solution. A survey of work in this synchronization technique is presented in (Krishna, 1990) and more detailed work can be found in (Baek *et al.*, 1994; Choi *et al.*, 1990; Ramanathan *et al.*, 1990b; Ramanathan *et al.*, 1990a; Shin and Ramanathan, 1988).

As a last remark, it should be noticed that this solution is an example of *frequency correction*. To actually synchronize clocks, some other mean must be used.

Software based clock synchronization A third option is to use the network connecting the computers to provide the clock synchronization service.

Its main advantage is to use the existing communication infrastructure for clock synchronization, and thus it does not require specific hardware. Suppressing additional hardware will eliminate (or, at the least, alleviate) problems caused by power consumption, size, price and interfacing that were posed by the preceding solutions. Using the network solves the problems of remote station visibility and requires no additional cabling. Finally, it is a solution that gives a great deal of freedom of choice to the system developer, as he/she can master the concepts and the technology used for clock synchronization (which are the same that are used to put the system to work).

Usually, the achievable precision and accuracy are worst than with the previous solutions. This solution also puts some extra charge on the site's CPU. Unless special hardware is used, the local

processor will be requested to execute the synchronization algorithm, on top of its normal activities. Network load increases, too. The use of special hardware for receiving the timing signals can be a solution in situations where tight synchronization is aimed for (see for instance (Hank, 1997; Kopetz and Ochsenreiter, 1987; Schossmaier *et al.*, 1997)).

Clock synchronization by means of message exchange has attracted the attention of researchers for some years now. The reasons for proposing new algorithms or new methods for their analysis have been, amongst others, fault-tolerance (Cristian *et al.*, 1986; Lamport and Melliar-Smith, 1985) and efficient implementations (Clegg and Marzullo, 1996; Drummond and Babaoğlu, 1993; Olson and Shin, 1994; Srikanth and Toueg, 1987). We can find also solutions and analysis for a specific type of systems: CAN (Controller Area Network, (Almeida *et al.*, 1999; Gergeleit and Streich, 1994; Hank, 1997)), FIP (Factory Instrumentation Protocol, (He *et al.*, 1990)), MAP (Manufacturing Automation Protocol, (Gora *et al.*, 1988)), UNIX (Gusella and Zatti, 1989) and Internet (Mills, 1991). (Kopetz and Ochsenreiter, 1987; Marzullo and Owicki, 1983; Verissimo and Rodrigues, 1992) propose other algorithms for clock synchronization. Surveys on this subject can be found on (He *et al.*, 1994; Mammeri and He, 1996; Simons *et al.*, 1989; Suri *et al.*, 1994) and (Attiya *et al.*, 1996; Dolev *et al.*, 1984; Fischer *et al.*, 1990; Lundelius and Lynch, 1984; Patt-Shamir and Rajsbaum, 1994) provide some fundamental results on clock synchronization by message exchange.

The vast majority of clock synchronization algorithms that have been presented belong to a class called *deterministic*. These algorithms suffer from a limitation: the attainable precision cannot be smaller than a value $\gamma = \epsilon(1 - 1/n)$, where $\epsilon = \Gamma_{\max} - \Gamma_{\min}$ is the uncertainty on the message delay, Γ , and n is the number of sites (Lundelius and Lynch, 1984). This result expresses two characteristics of deterministic clock synchronization algorithms: i) the attainable precision has a non-null lower bound; ii) there must be an upper bound on the message delay (Γ_{\max} must be finite). Another class of clock synchronization algorithms, *non-deterministic* clock synchronization algorithms, strive at circumventing this limitation, by offering a precision that can be as small as desired. The price to pay is a (small) probability that the system will fail to synchronize. In general, this probability of failure can be made as small as desired by sending a sufficient number of messages. These properties have been exploited in algorithms such as those presented in (Arvind, 1994; Cristian, 1989; Cristian and Fetzer, 1994; Le Lann, 1990; Olson and Shin, 1991; Rangarajan and

Tripathi, 1991) and a comparative study can be found in (Fonseca, 1999).

The methods for synchronizing clocks we have presented (based on the dissemination of clock values by hardware, radio transmission and message exchange) of are not mutually exclusive. They can be used together and examples of this can be found in (Schmid, 1995; Veríssimo *et al.*, 1997).

4. CONCLUSION

Clock synchronization allows the establishment of a global time base, a requirement for many distributed systems applications. The properties that can be expected from clocks in a computer system were presented and formalized. Some of these properties are not present in the physical clock, and they must be enforced by means of a clock synchronization algorithm. A classification for clock synchronization solutions was presented, with the purpose of facilitating the choice of a clock synchronization solution for a specific system.

5. REFERENCES

- Almeida, Luís, Pedro Fonseca, José Alberto Fonseca and Zoubir Mammeri (1999). Scheduling and clock synchronization in CAN-based distributed systems. In: *Proceedings 6th International CAN Conference*. CAN in Automation. Turin, Italy. pp. 5–02–5–10.
- Arvind, K. (1994). Probabilistic clock synchronization in distributed systems. *IEEE Trans. Parallel and Distributed Systems* **5**(5), 474–487.
- Attiya, Hagit, Amir Herzberg and Sergio Rajsbbaum (1996). Optimal clock synchronization under different delay assumptions. *SIAM J. Comput.* **25**(2), 369–389.
- Baek, Yunju, Heung-Kyu Lee and Kiyel Ryu (1994). A new hardware based fault-tolerant clock synchronisation scheme for real-time multiprocessor systems. *Microelectronics and reliability* **34**(2), 335–349.
- Choi, Bong-Rak, Kyn Ho Park and Myunghwan Kim (1990). An improved hardware implementation of the fault-tolerant clock synchronization algorithm for large multiprocessor systems. *IEEE Transactions on Computers* **C-39**(3), 404–407.
- Clegg, Matthew and Keith Marzullo (1996). Clock synchronization in hard real-time distributed systems. Technical Report CS96-478. University of California, San Diego.
- Cristian, F. (1989). Probabilistic clock synchronization. *Distributed Computing* **3**, 146–158.
- Cristian, Flaviu and Christof Fetzer (1994). Probabilistic internal clock synchronization. In: *Proc. 13th Symposium on Reliable Distributed Systems*. IEEE. pp. 22–31.
- Cristian, Flaviu, Houtan Aghili and Ray Strong (1986). Clock synchronization in the presence of omission and performance faults, and processor joins. In: *Proc. Intern. Conf. on Fault-Tolerant Computing*. pp. 218–223.
- Dana, Peter H. (1997). Global positioning system (GPS) time dissemination for real-time applications. *Real-Time Systems* **12**(1), 9–40.
- Dolev, D., H. Halpern and R. Strong (1984). On the possibility and impossibility of achieving clock synchronization. In: *Proc. 16th ACM STOC*. pp. 504–511.
- Drummond, Rogério and Özalp Babaoglu (1993). Low-cost clock synchronization. *Distributed Computing* **6**, 193–203.
- Fischer, Michael J., Nancy A. Lynch and Michael Merrit (1990). *Easy Impossibility Proofs for Distributed Consensus Problems*. pp. 147–170. in Simons and Spector (1989).
- Fonseca, Pedro (1999). Modélisation et validation des algorithmes non-déterministes de synchronisation des horloges. PhD thesis. IN-PL/UA. Nancy, France/Aveiro, Portugal.
- Fonseca, Pedro, José Alberto Fonseca and Zoubir Mammeri (1998). Can we trust Internet? a case study on clock synchronization algorithms. In: *Proceedings of INDC'98 — 7th I-FIP/ICCC Conference on Information Networks and Data Communications*. Aveiro, Portugal. pp. 69–82.
- Frerking, Marvin E. (1978). *Crystal Oscillator Design and Temperature Compensation*. Van Holland Reinhold.
- Gergeleit, Martin and Hermann Streich (1994). Implementing a distributed high-resolution real-time clock using the CAN-bus. In: *Proceedings of the 1st International CAN-Conference*. CAN in Automation. Erlangen.
- Gora, W., O. Hersos and S. K. Tripath (1988). Clock synchronization in the factory floor. *IEEE Transactions on Industrial Electronics* **35**(3), 372–380.
- Gusella, R. and S. Zatti (1989). The accuracy of the clock synchronization achieved by TEMPO in Berkeley Unix 4.3 BSD. *IEEE Trans. on Software Engineering* **15**(7), 847–853.
- Hank, Peter (1997). PeliCAN: A new CAN controller supporting diagnosis and system optimization. In: *Proceedings of the 4th International CAN Conference*. pp. 4–12 – 4–18.
- He, Jiying, Zoubir Mammeri and Jean-Pierre Thomesse (1994). Modélisation des techniques de synchronisation d'horloges. *Technique et science informatiques* **13**(2), 185–221.

- He, Jying, Zoubir Mammeri and Jean-Pierre Thomesse (1990). Clock synchronization in real-time distributed systems based on FIP field-bus. In: *Proc. of the 2nd IEEE Conference on Dist. Comp. Syst.*. IEEE. Cairo. p-p. 135–141.
- Kopetz, Herman and Wilhelm Ochsenreiter (1987). Clock synchronization in distributed real-time systems. *IEEE Trans. on Computers* **C-36**(8), 933–940.
- Krishna, C. M. (1990). Fault-tolerant synchronization using phase-locked clocks. *Microelectronics and Reliability (UK)* **30**(2), 275–287.
- Lampert, L. and P. M. Melliar-Smith (1985). Synchronizing clocks in the presence of faults. *Journal of the Association for Computing Machinery* **32**(1), 52–78.
- Le Lann, Gérard (1990). Synchronisation statistique d'horloges physiques: étude algorithmique. Technical report. INRIA.
- Lichtenecker, Reiner (1997). Terrestrial time dissemination. *Real-Time Systems* **12**(1), 41–61.
- Lundelius, Jennifer and Nancy Lynch (1984). An upper and lower bound for clock synchronization. *Information and Control* **62**, 190–204.
- Mammeri, Z. and J. He (1996). Modeling and timing performance analysis of deterministic clock synchronization algorithms. In: *Proc. 9th Int. Conference on Parallel and Distributed Systems*. Dijon, France. pp. 219–224.
- Marzullo, Keith and Susan Owicki (1983). Maintaining time in a distributed system. In: *Proceedings of the Second ACM Symposium on Principles of Distributed Computing*. pp. 44–54.
- Mills, David L. (1991). Internet time synchronization: The network time protocol. *IEEE Trans. on Commun.* **COM-39**(10), 1482–1493.
- Mills, David L. and Ajit Thyagarajan (1994). Network time protocol version 4 - proposed changes. Technical Report 94-10-2. Electrical Engineering Department, University of Delaware.
- Olson, A. and K. G. Shin (1991). Probabilistic clock synchronization in large distributed systems. In: *Proc. Distributed Computing Systems*. pp. 290–297.
- Olson, A. and K. G. Shin (1994). Fault-tolerant clock synchronization in large multicomputer systems. *IEEE Trans. Parallel and Distributed Systems* **5**(9), 912–923.
- Patt-Shamir, Boaz and Sergio Rajsbaum (1994). A theory of clock synchronization. In: *Proceedings of the 26th ACM Symposium on Theory of Computing*. ACM. Montreal, Quebec, Canada. pp. 810–819.
- Ramanathan, P., Dilip D. Kandhur and Kang G. Shin (1990a). Hardware assisted software clock synchronization for homogeneous distributed systems. *IEEE Trans. on Computers* **C-39**(4), 514–524.
- Ramanathan, Parameswara, Kang. G. Shin and Ricky W. Butler (1990b). Fault-tolerant clock synchronization in distributed systems. *IEEE Computer* **23**(10), 33–42.
- Rangarajan, S. and S. K. Tripathi (1991). Efficient synchronization of clocks in a distributed system. In: *Proceedings of Real-Time Systems Symposium*. pp. 22–31.
- Schmid, Ulrich (1995). Synchronized universal time coordinated for distributed real-time systems. *Control Engineering Practice* **3**(6), 877–884.
- Schossmaier, Klaus, Ulrich Schmidt, Martin Horauer and Dietmar Loy (1997). Specification and implementation of the universal time coordinated synchronization unit (UTCSU). *Real-Time Systems* **12**(1), 295–328.
- Shin, Kang G. and P. Ramanathan (1988). Transmission delays in hardware clock synchronization. *IEEE Transactions on Computers* **37**(11), 1465–1467.
- Simons, B. and Spector, A., Eds.) (1989). *Fault Tolerant Distributed Computing*. number 448 In: *Lecture Notes in Computer Science*. Springer-Verlag.
- Simons, B., J. Lundelius-Welch and N. Lynch (1989). An overview of clock synchronization. pp. 84–96. In: Simons and Spector (1989).
- Srikanth, T. and S. Toueg (1987). Optimal clock synchronization. *Journal of the Association for Computing Machinery* **34**, 626–645.
- Sterzbach, Bernhard (1997). GPS-based clock synchronization on a mobile, distributed real-time system. *Real-Time Systems* **12**(1), 63–76.
- Suri, Neeraj, Michelle M. Hugue and Chris J. Walter (1994). Synchronization issues in real-time systems. *Proceedings of the IEEE* **82**(1), 41–54.
- Verissimo, P. and L. Rodrigues (1992). A posteriori agreement for fault-tolerant clock synchronization on broadcast networks. In: *Proc. of FTCS*. pp. 527–536.
- Verissimo, Paulo, Luís Rodrigues and António Casimiro (1997). CesiumSpray: A precise and accurate global time service for large-scale systems. *Real-Time Systems* **12**(1), 243–294.
- Vig, John R. (1992). Introduction to quartz frequency standards. Technical Report SLCET-TR-92-1. US Army Communications-Electronics Command.
- Wannenmacher, M. and W. A. Halang (1994). GPS-based timing and clock synchronization for real time computers. *Electronics Letters* **30**(20), 1653–1654.