Redes de Comunicação em Ambientes Industriais
Aula 9

Luís Almeida lda@det.ua.pt
Paulo Pedreiras pedreiras@det.ua.pt

Electronic Systems Lab-IEETA / DET
Universidade de Aveiro
Aveiro, Portugal

In the previous episode ...

Traffic scheduling

- ✓ Establishes the **relative order** of the message transmissions
- ✓ **Constraints by the MAC**
  - ✓ **Minimum tx period (e.g. TDMA), jitter (e.g. Token), dead interval (pooled systems)**
- ✓ **On-line/off-line (table based), static/dynamic**
- ✓ **Resemblances with task scheduling (in CPUs)**
  - ✓ **CPU / Bus, Tx time / Execution time, ...**
  - ✓ **Preemption allowed only with multi-packet messages**
- ✓ **Similarities with server scheduling**
  - ✓ **Fraction of the bandwidth allocated to each node**

In the previous episode ...

- ✓ Scheduling criteria:
  - ✓ Fixed priorities (RM, DM, importance/value)
  - ✓ Dynamic priorities (EDF, LLF, FCFS)
- ✓ Schedulability analysis:
  - ✓ Utilization
  - ✓ Response time
  - ✓ Timeline
  - ✓ Branch and bound (for static table/based)

WorldFIP
Factory Instrumentation Protocol

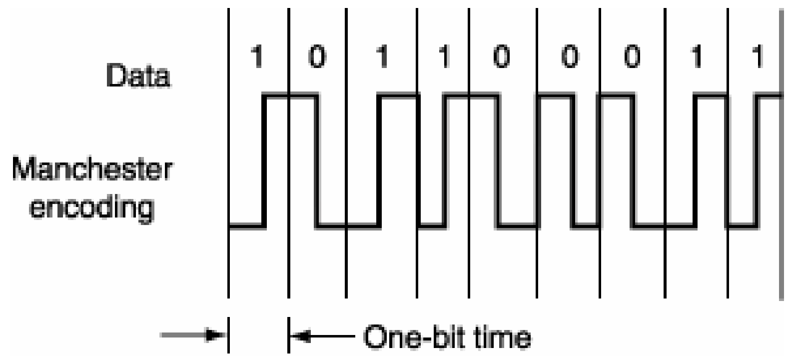
www.worldfip.org

WorldFIP

- ✓ Created, in the 80s, in France for use in **process control** and **factory automation**.
- ✓ AFNOR standard C46601..7 (89-92)
- ✓ CENELEC standard EN50170,vol.3 (96)
- ✓ IEC standard 61158, type 7 (2000)
- ✓ Typical in train control systems

WorldFIP

- ✓ **Broadcast** serial **bus**
- ✓ Synchronous transmission
- ✓ Manchester encoding



- ✓ Transmission rates **32 Kbit/s**, **1 Mbit/s** and **2.5 Mbit/s** on copper or **5 Mbit/s** on optical fiber
- ✓ Maximum Length 2000m @ 1Mbit/s
- ✓ Max. number of nodes 256

WorldFIP

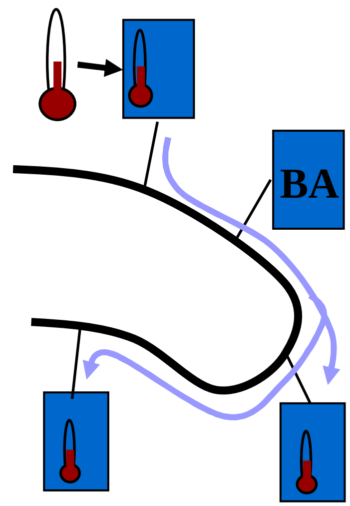
2 messaging systems:

- ✓ **MPS – real-time** services, periodic, aperiodic;
  MMS subset – non-real-time messaging

- ✓ Data **payload** between **0 and 128 bytes** (256 for non real-time messages)
- ✓ **Source-addressing** (message identifiers with 16 bits)
- ✓ **Master-Slave** bus access control
  - ✓ BA - Bus Arbitrator / Distributor

WorldFIP

- ✓ **MPS** – *Messagerie Periodique e Sporadique*
- ✓ **Producer-Distributor-Consumer** model
- ✓ Concept of **Network Variable**
  - ✓ Entity that is distributed (several local copies coexist in different nodes)
  - ✓ Can be **periodic** or **aperiodic**
  - ✓ Local copies of **periodic variables** are **automatically refreshed** by the network
  - ✓ Local copies of **aperiodic variables** are refreshed by the network upon **explicit request**



WorldFIP

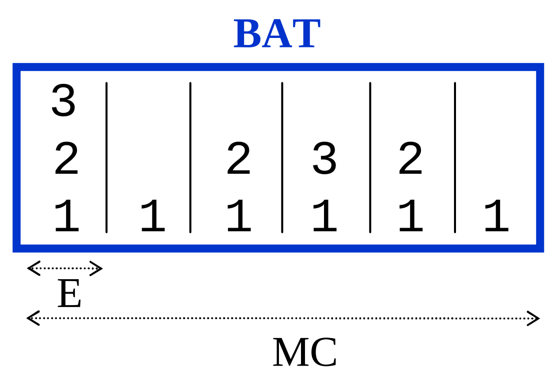
- **Table-based** scheduling of **periodic** traffic
- Table (BAT) organized in **cycles**
  - **Elementary cycles**
    duration E (=GCD of periods)
    Macro-cycles
    LCM of periods (in ECs)
- Scheduling model

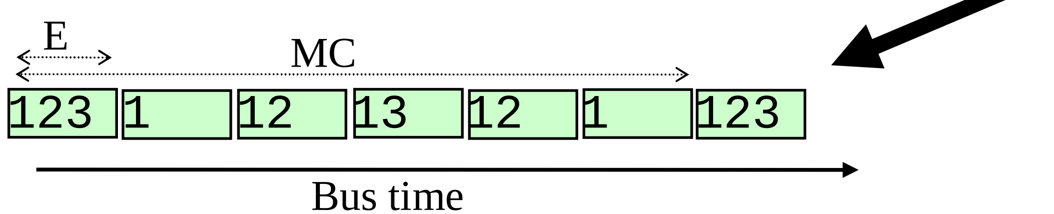
  $\Gamma_p \equiv \{v_i : v_i(C_i, T_i, D_i, O_i), i=1..N_p\}$

  $i=1..N_p$, $C_i << E$, T_i, O_i integer mult. of E

Periodic Variables:

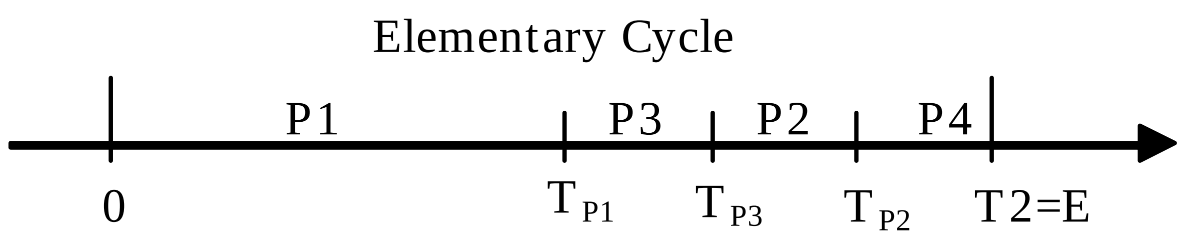
i	1	2	3
P_i	1	2	3





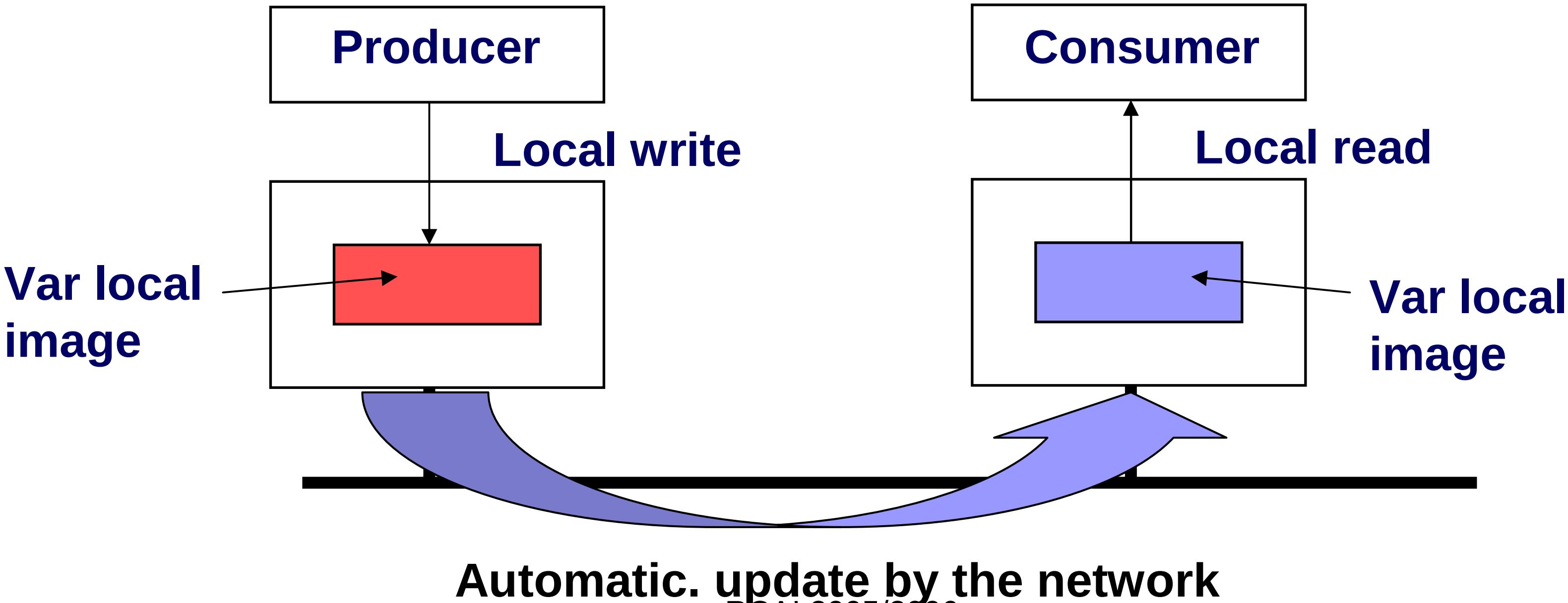
WorldFIP

- Elementary cycles organized in **phases**:
  - **Periodic** (P1)
  - **Aperiodic** (P3)
  - MMS **messages** (P2)
  - Sync - **padding** (P4)



WorldFIP

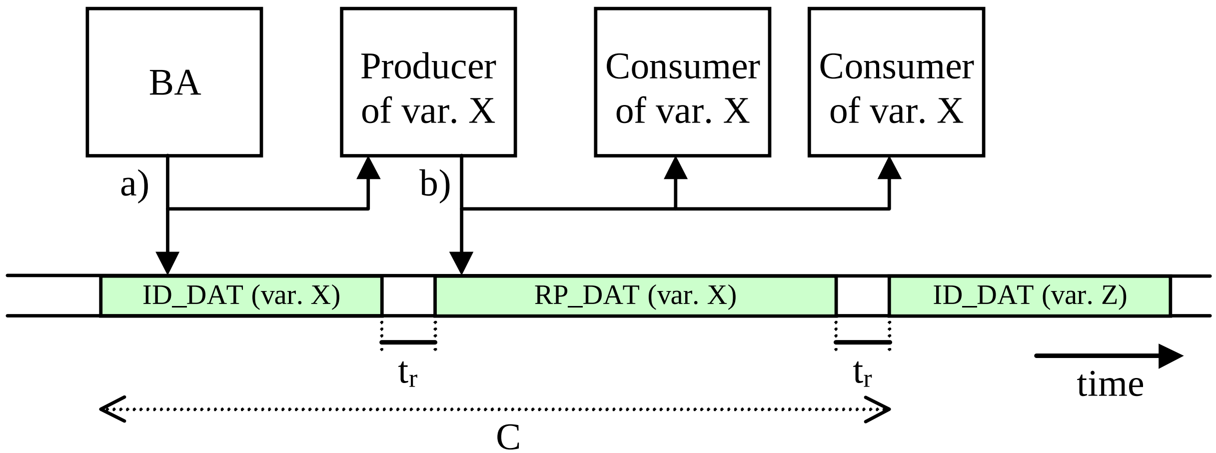
- **Buffer transfer** (elementary transaction)
  - Local read and write are carried out in local buffers and are independent of the bus activity
  - Consumer(s) local buffers are automatically updated by the network



WorldFIP

- ✓ **Buffer transfer** (elementary transaction)

(figure by Tovar)



$$C = \frac{len(\mathrm{ID\_DAT}) + len(\mathrm{RP\_DAT})}{tx\_rate} + 2 \times t_r$$

WorldFIP

- **Data efficiency** (periodic transfers)

t_r = **turn around** time, 10-70 tmacs (bit times)
len(ID_DAT) = 64 bits
len(RP_DAT) = 48 + data bits

Must give time for the slowest node to decode the master messages and answer when addressed

$$\text{Data\_eff} = \frac{\text{data bits}}{\text{data bits} + 64 + 48 + 2 * t_r}$$

E.g.

Tr (tmacs)	Data bits	Data eff. (%)
20	16	9.5%
20	64	30%
20	1024	87%

WorldFIP

- **Schedulability analysis**
  - **Just build the table (BAT) !**
  - Typically using **branch and bound** techniques to optimize the schedule (e.g. wrt to jitter of periodic buffer transfers, precedence and window constraints)
  - The BAT can also be built using common criteria such as fixed priorities, or EDF

WorldFIP

- **Building the BAT with common criteria**
  - EC by EC, using **fixed priorities**

For each EC scan variables

```
{BAT_{m,n}=0 for all m and n}
for (k=1;k≤N_p;k++){δ_{k,1}=0;}
for (n=0;(n ≤ LCM(T));n++){
        Load_n=0;
        m=0;
        for (k=1;k≤N_p;k++){
            δ_{k,n+1}= δ_{k,n};
            if (Load_n + δ_{k,n}*C_k <= E) {
                Load_n = Load_n + δ_{k,n}*C_k;
                m++;
                BAT_{m,n} = k;
                δ_{k,n+1}=0;
                }
            if (n mod T_k/E=O_k) δ_{k,n+1}=1;
            }
        }
```

Clear the initial BAT

Go through all ECs up to end of table

Search variables in fixed priority order

Place variable in the table

Periodic activations with offsets

WorldFIP

- **Building the BAT with common criteria**
  - EC by EC, using **earliest deadline**

Same algorithm can still be used with EDF

Only difference is the extra sorting in line 5

Sort only when new vars become ready

```
1    {BAT_{m,n}=0 for all m and n}
2.  for (k=1;k≤N_p;k++){δ_{k,1}=0;}
3.  for (n=0;(n ≤ LCM(T));n++){
4.         Load_n=0; m=0;
5.         {sort vars by increasing
            distance to deadline}
6.         for (k=1;k≤N_p;k++){
7.             δ_{k,n+1}= δ_{k,n};
8.             if (Load_n + δ_{k,n}*C_k <= E) {
9.                Load_n = Load_n + δ_{k,n}*C_k;
10.               m++;
11.               BAT_{m,n} = k;
12.               δ_{k,n+1}=0;
13.               }
14.            if (n mod T_k/E=O_k) δ_{k,n+1}=1;
15.            }
16.        }
```

Clear the initial BAT

Go through all ECs up to end of table

Scan all variables in EDF order

WorldFIP

- **Building the BAT with common criteria**
  - For fixed priorities it's more efficient var by var

For each Var place it in all due ECs in the table

More efficient when the table is lightly to moderately loaded

```
{BAT_m,n=0 for all m and n}
{Load_n=0 and m_n=0 for all n}
for (k=1;k≤N_p;k++){
        for (n=O_k;(n≤LCM(T));n=n+T_k){
            j=n mod LCM(T);
            while (Load_j+C_k > E) {
                j= ++j mod LCM(T);
                 {check if deadline missed}
                }
            Load_j=Load_j+C_k;
            m_j++;
            BAT_mj,n = k;
            }
        }
```

Clear the initial BAT and related vectors

Go through all vars in priority order

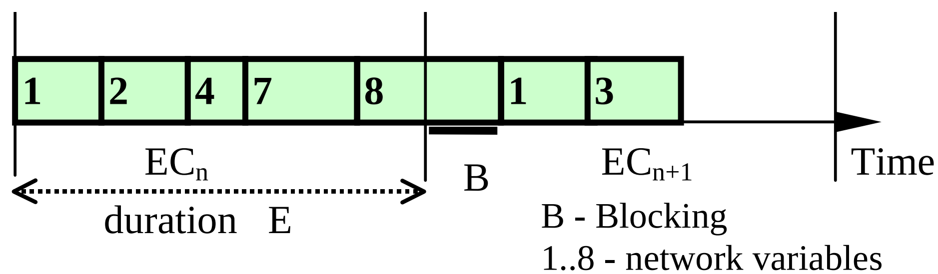
For each var go through all instances

Place variable in the table

WorldFIP

- **Schedulability analysis**
  - But **on-line scheduling** is also possible
  - Common analysis can be useful
    - Need to account for **blocking** due to non-preemption

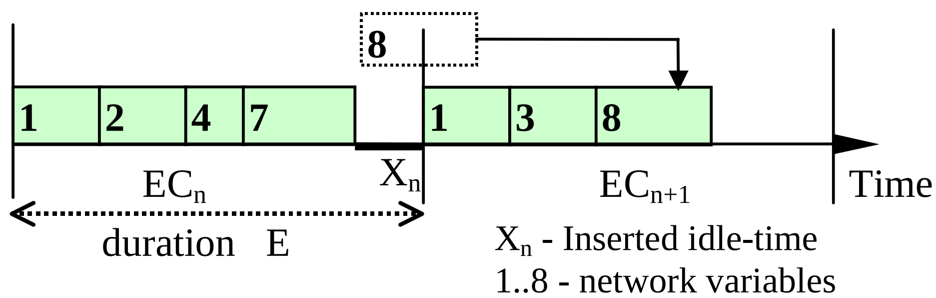
$$B_i = \max_{l=j..N} (C_l)$$ j is first var that can cause blocking



- RM: $$\sum_{1}^{N} \frac{C_i}{T_i} + \max_{1..N}\left(\frac{B_i}{T_i}\right) < N(2^{1/N} - 1)$$
- EDF: $$\sum_{1}^{N} \frac{C_i}{T_i} + \max_{1..N}\left(\frac{B_i}{T_i}\right) < 1$$ (as in SRP, Stack Resource Protocol)

WorldFIP

- **Schedulability analysis**
  - Or you can get rid of the blocking using inserted idle-time

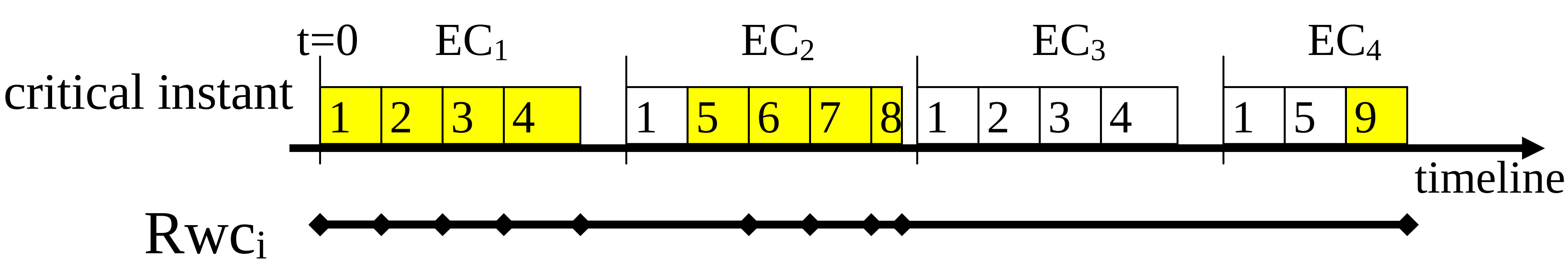


$$\boxed{C_i' = C_i * \frac{E}{E - X_{max}}}$$

  - In this case, you can use any analysis for preemptive scheduling with

WorldFIP

- **Schedulability analysis**
  - Or you can build the initial part of the table after the critical instant (timeline analysis)
  - For fixed priorities, first occurrence after critical instant is the one with longest delay wrt periodic release
  - Consider the following set of 9 variables with periods given by T_1=1, $T_{2..5}$=2, $T_{6..9}$ >3



WorldFIP

✓ Schedulability analysis

✓ Algorithm for the timeline analysis

Rwc – Worst-case response time for each variable
Builds the table, EC by EC until all variables are scheduled once

Same as for building the BAT EC by EC

```
for (k=1;k≤N_p;k++){Rwc_k=0; δ_k,1=1;}
for (n=1;(n ≤ ⌈D_Np/E⌉ and Rwc_Np=0);n++){
Load_n=0;
    for (k=1;k≤N_p;k++){
        δ_k,n+1= δ_k,n;
    if (Load_n + δ_k,n*C_k <= E) {
         Load_n = Load_n + δ_k,n*C_k;
         δ_k,n+1=0;
         if (Rwc_k=0) Rwc_k=(n-1)*E+Load_n;
         }
        if (n mod T_k/E=0) δ_k,n+1=1;
        }
    }
}
```

Cycle to scan EC by EC up to placing last variable

Cycle to choose next variable

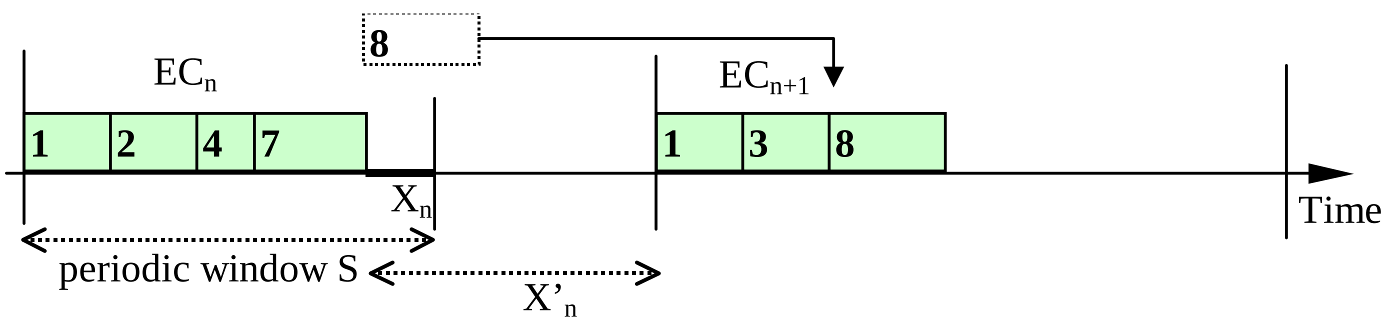
Place variable in the EC if space

Calculate Rwc if not done yet

Account for new instances of higher priority variables

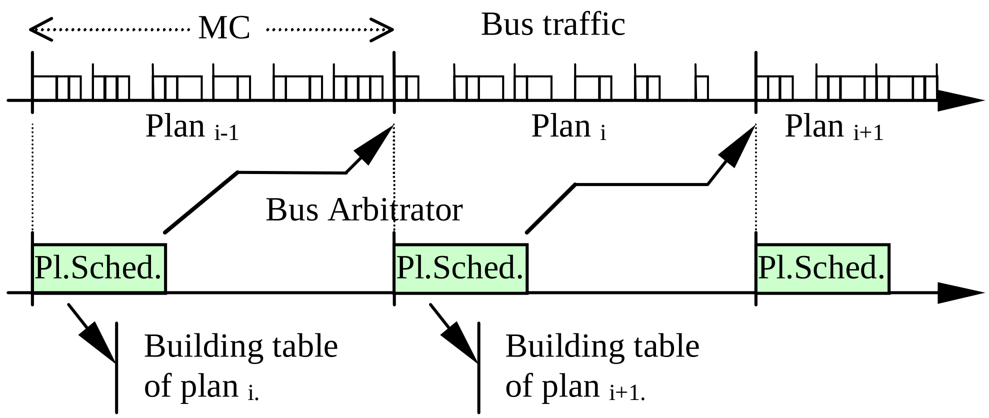
WorldFIP

- **Schedulability analysis**
  - You can also constrain the periodic traffic to a smaller periodic window
  - Using the inserted idle-time approach $X'_{max} = E - (S-X_{max})$



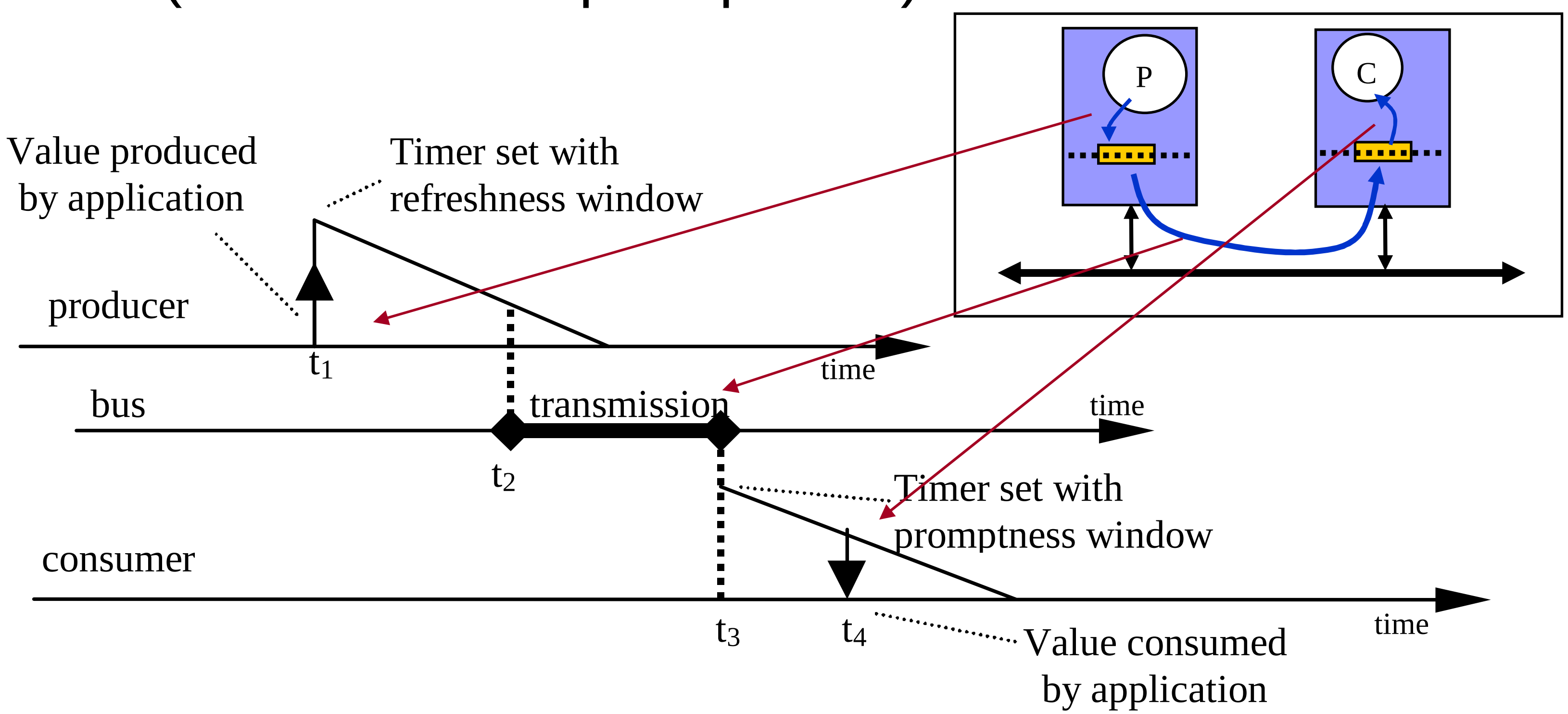
WorldFIP

- Constraining the **size** of the schedule **table**
  - Size given by LCM(T_i)… can be very large!
  - Make all periods **harmonic**
    - i=1..N, $T_i = A^{k_i} * T$, k_i integer or null, A and T constants
  - Bound table length and **rebuild** it **on-line**
    (planning scheduler)



WorldFIP

- Delivers information on **temporal accuracy** (refreshness & promptness)



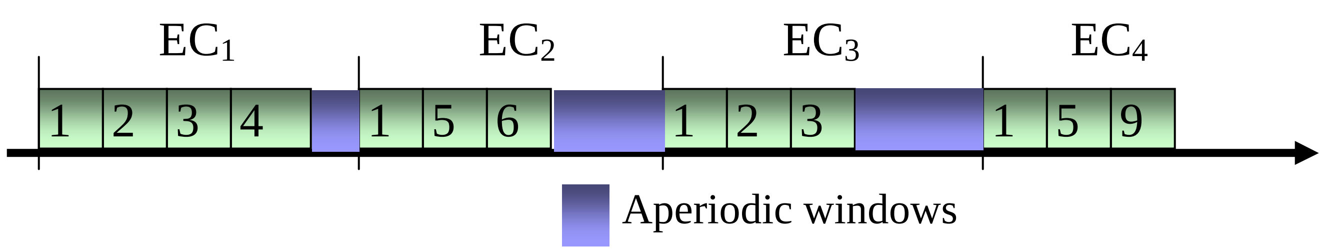
WorldFIP

- ✓ **Aperiodic message transmission**

Shared Dynamic Window

- ✓ **Multiple phase** cycle
- ✓ **Several nodes** can transmit in that window
- ✓ The window **width varies dynamically**
  (uses the part of a cycle not used by periodic traffic)

- ✓ Nodes use the normal **periodic transfers** to **signal** the presence of a**periodic requests**
  (using one control bit in the RP_DAT frame).

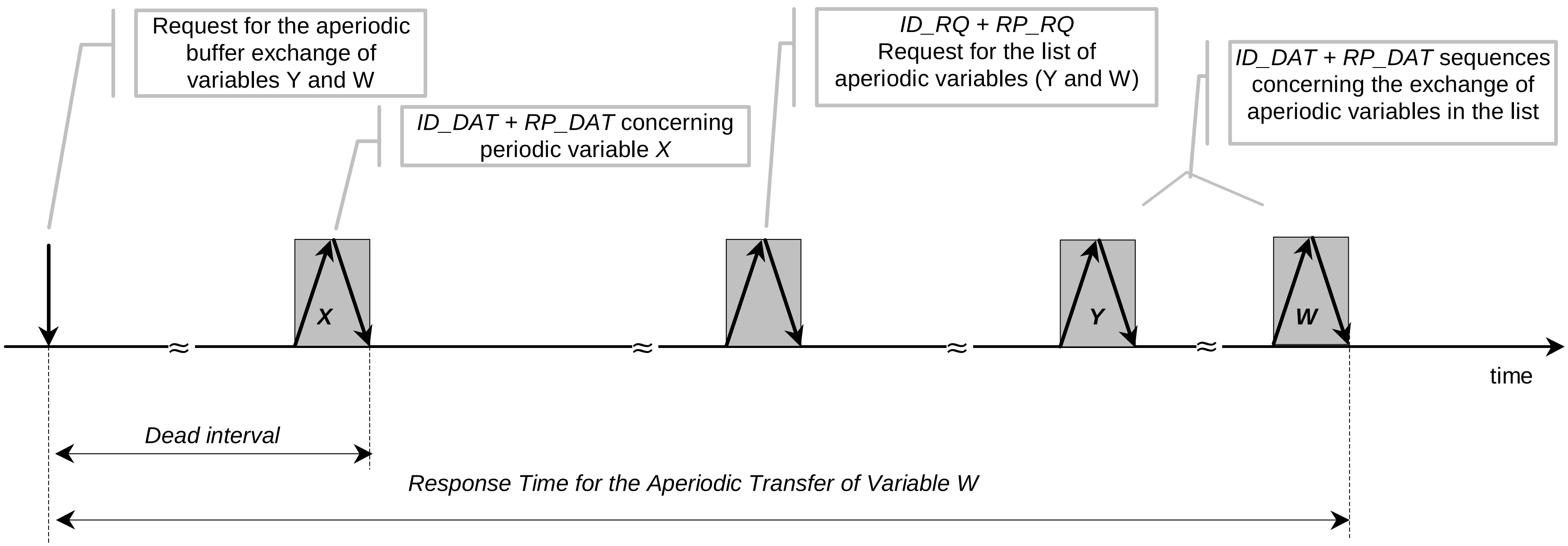


WorldFIP

- ✓ The requests are stored in two **FIFO** queues with different priorities, **urgent** and **normal**
- ✓ Then the **master** directly **polls** the nodes that requested service, asking for the **identification** of the requested aperiodic transfers
(list request – **ID_RQ / RP_RQ**)
- ✓ Finally, the master handles **each** requested **aperiodic transfer** as a **regular** buffer transfer (**ID_DAT / RP_DAT**)
- ✓ All these transfers are carried out within one or more consecutive aperiodic windows

WorldFIP

✓ Aperiodic requests handling



Sequence of network transfers from the same node to transmit aperiodic variables Y and W

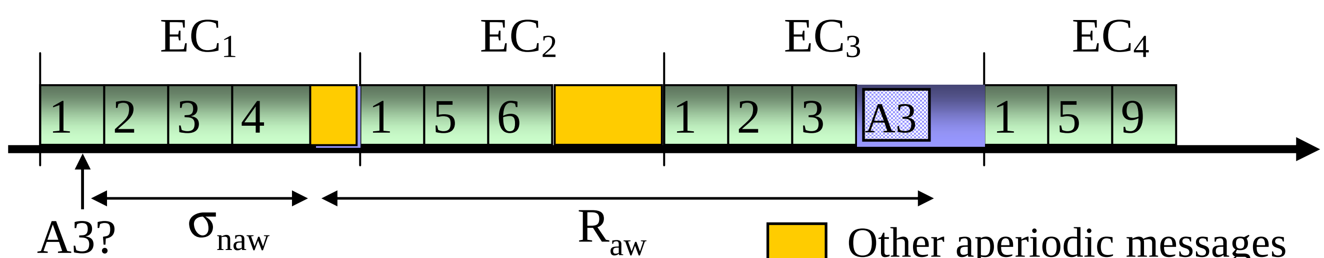
WorldFIP

✓ **Response-time to an aperiodic tx request (R_a)**

$$R_a < \sigma_{naw} + R_{aw}$$

σ_{naw}=longest consecutive period of time without access to the aperiodic window – **dead interval**

R_{aw}=response time from start of the window in which the request is first considered (possibly extending to following windows)



WorldFIP

- **Example:**
  - **Aperiodic window** is the **time left** after the periodic traffic in each EC

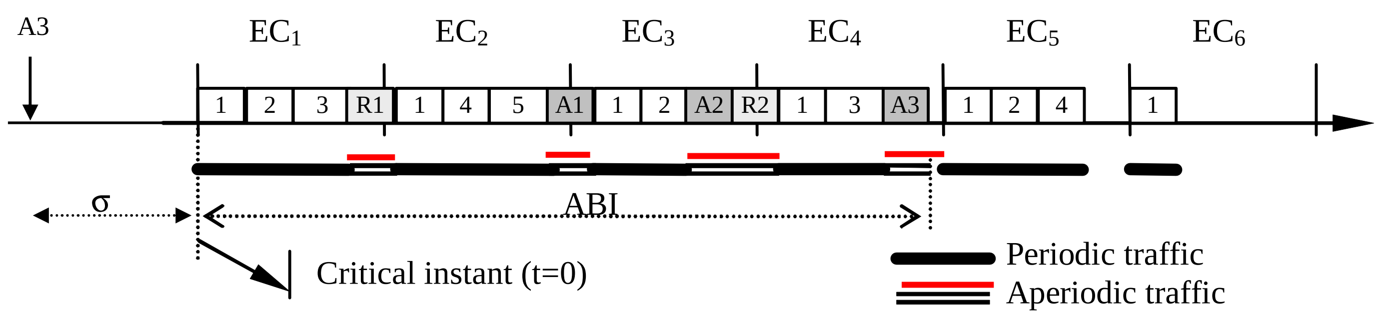
$$R_a < \sigma_{naw} + \mathrm{len(ABI)}$$

<u>Periodic variables (C,P):</u>
1- (250µs,1ms)
2- (250µs,2ms)
3- (300µs,3ms)
4- (250µs,4ms)
5- (300µs,6ms)

E=1ms

<u>Aperiodic variables (node,Ca):</u>
1- (Node 1,200µs)
2- (Node 1,250µs)
3- (Node 2,200µs)

(Cl_1 = 250µs)
(Cl_2 = 200µs)



WorldFIP

- **Performance of the aperiodic system:**
  - If **no minimum width** is guaranteed, then this method behaves like a **background server**
  - However, a **minimum width** can be set, which **guarantees a minimum bandwidth** to handle the aperiodic traffic.
  - This is a mixed scheme that results in **faster response** time but at the price of **lower efficiency** due to the static bandwidth allocation

Summary:

- ✓ IEC standard 61158, type 7 (2000)
- ✓ Typical in train control systems
- ✓ Periodic and aperiodic traffic
- ✓ Producer/Distributer/Consumer cooperation model
- ✓ Periodic traffic: table based scheduling (BAT)
- ✓ Building the BAT:
  - ✓ any scheduling policy possible
  - ✓ BAT size may be a problem (LCM)

Summary:

- ✓ On-line scheduling possible
  - ✓ Admission control using adapted analysis (e.g. RM, EDF) or timeline analysis
- ✓ Data temporal validity (promptness and refreshness)
- ✓ Aperiodic requests handled in a shared dynamic window
  - ✓ Use the time left by periodic messages
  - ✓ Signalisation of aperiodic requests piggybacked in periodic messages
  - ✓ Pooled by the Distributor node
  - ✓ WCRT computation possible (dead interval + asynchronous busy window)