Slow timing over ‘common purpose’ network media

An alternative to slow timing over a dedicated timing network

Requirements

- Guaranteed delivery of multi-cast messages to allow a set of clients that to undertake a synchronised action.
- Message are emitted from a master process which can execute synchronised message sequences, and/or send individual message on request of other applications.
- Good responsiveness: short latency between the need for message transmission and execution of the action.

synchronised action

Synchronised action can be accomplished in two ways:

- The message itself is time synchronised
  - i.e. the SPS and LEP MTG events are synchronised to the ms slot in which they are transmitted.
- The message encodes the time synchronisation for some time in the future:
  - at the next top do this (p.s. telegram style)
  - at time T do this (possible LHC style?)

  Note: Message will need some extra warning time.

At time T do this...

Lets synchronise our watches, Gary...

- If the timing receiving equipment have their own synchronised internal clocks, then the message distribution does not need to be synchronised any more.
- But we still need a guaranteed maximum latency.

And what if the transmission is unreliable?

Need a protocol, adding more latency…

But is the problem solvable ?

The general dilemma

- Two generals plan a simultaneous attack on there adversary. They will only succeed if they both attack at the same time. If one of them would attack alone, it will be a disaster…

Unfortunately, they have a rather unreliable communication system based on pigeons.

And with too many pigeons in the air, there is a finite possibility that they collide.

The general dilemma (2)

General A sends a message to his colleague:

- I will attack tomorrow morning at 7:30 provided you are ready as well. Please confirm.

The general dilemma (3)

General B receives the message and thinks great, we go for it. So he answers:

- I am ready to attack tomorrow morning at 7:30.

  But then he thinks, what if this message does not arrive? I will attack alone! So he adds:

- Please confirm this message.

The general dilemma (4)

General A receives the answer and thinks “it is gonna happen”. So he writes:

- Got your message, see you tomorrow.

  But then he also thinks, what if this message does not arrive? B will not come! So he adds:

- Please confirm this message.

The general dilemma (n)

General B (and then A, and then B, ….) receives the answer and replies:

- Got your message, see you tomorrow.

  But then he thinks again, what if this message does not arrive? So he adds:

- Please confirm this message.

Accelerator equipment dilemma

- Starting a ramp, making a trim, or performing any serious action will only succeed if all equipment involved in the action, participate.
- If one of them would fail, it could be a mess.

Note: The situation is slightly different from the general dilemma, because here it is the “pentagon” talking to the generals.

Timing based on unreliable communication

- A central system may broadcast messages over an unreliable network provided that there is
- feedback
- a recovery algorithm

Registration

- The master needs to know who its clients are.

  This is different from the current MTG situation: The MTG broadcasts and does not need to care who listens.

- A client that want to listen to a given broadcast message, must register himself with the master and specify:
  - Message identifier
  - Required warning time
  - The subscription type
    - One shot
    - Permanent
  - The acknowledge parameters
    - timeout time (if set to zero, then no acknowledge is required)
    - Number of times to repeat a message on acknowledge timeout
    - abort transaction on acknowledge timeout
    - severity, to decide what to do in case of a missing acknowledge (e.g.dump beam)

Message distribution

1. When a master needs to multi-cast a message, it will inform all its slaves that have registered for the message and wait for their response.
2. If all the slaves replied to the master and accepted the message, the case is closed.
    3. However, if one slave did not reply, or rejected the message, the master will send an abort.
    4. If all slaves replied (including the trouble maker), the action is aborted.

If any slave did not reply to the abort we are in deep trouble. The master better dumps the beam!

And how reliable is its connection with the beam dump?

Network down = dump the beam

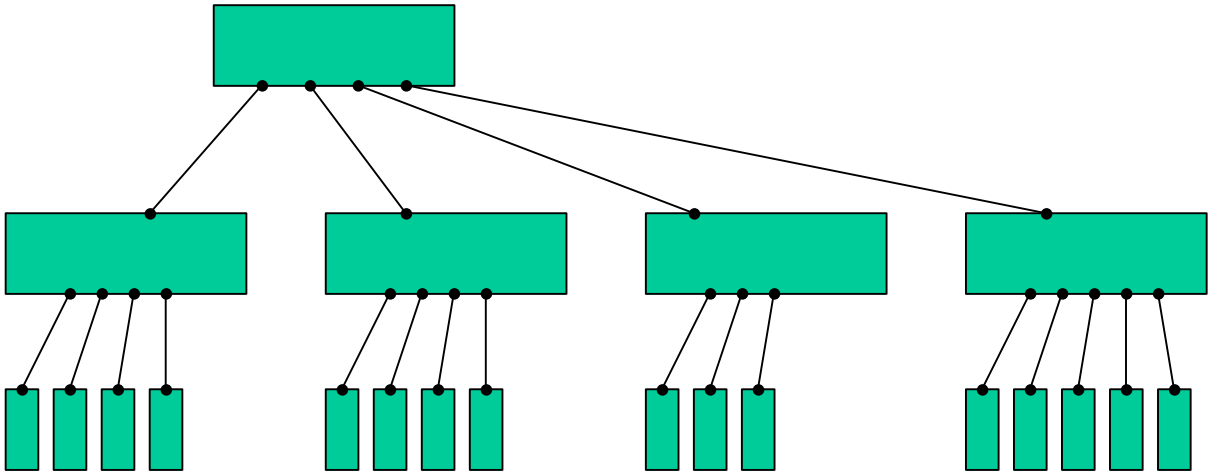
What if the network connection to a slave is down:

- The master will send a message to its slaves.
- All but one slave will answer.
- The master will send an abort.
- All but the same slave will acknowledge.
- The master does not know in what state the slave is and it will dump the beam.

The master and slaves keep track of the communication state and will not undertake anything serious in case of malfunctioning.

Hierarchical organisation

- A single master talking to >1000 end clients with point to point communication?
- Install repeater processes and build a hierarchy
- One - N repeater processes per IP, each on talking to N’ clients



- A repeater is made from master and a slave process.
- A repeater could also reside on an equipment bus and use a different communication medium with its slaves as with its master