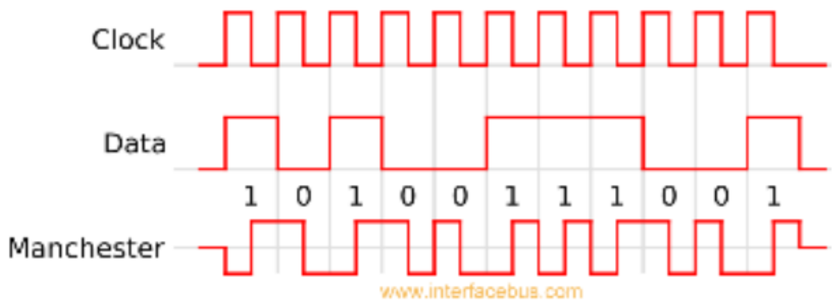
Exam 2016 - solution

Problem 1

a) **DC**-balance (equal number of ‘0’ and ‘1’ to get zero offset) and **clock recovery** (Need transitions between ‘0’ and ‘1’ to recover the clock).

b) The data and clock are combined into one signal, so that the receiver can recover the transmitter clock (self-clocking). XOR of data and clock gives at least one transition for each clock period; '1' is a low to high transition and '0' is a high to low transition.

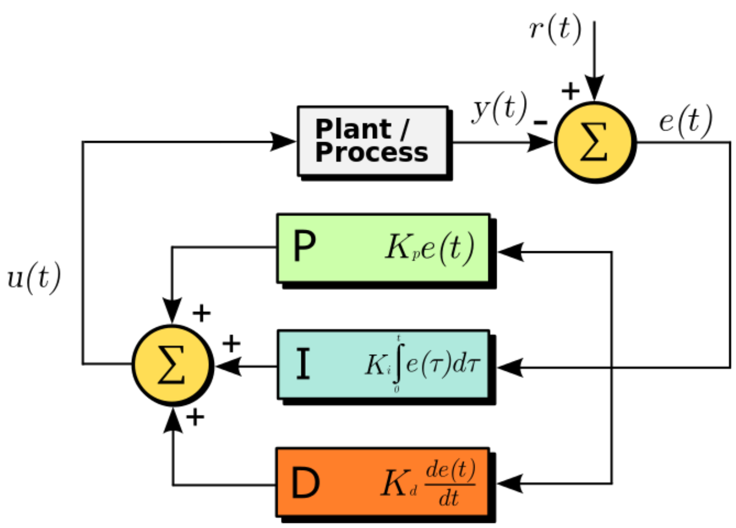
c) Detect the rising and falling edges of the Manchester encoded signal, but ignore the rising/falling edges with a smaller separation than one clock period (should only get one level transition pr. clock period).



d) A controller that switches between two states; e.g. either completely on or completely off. Examples: residential thermostats.

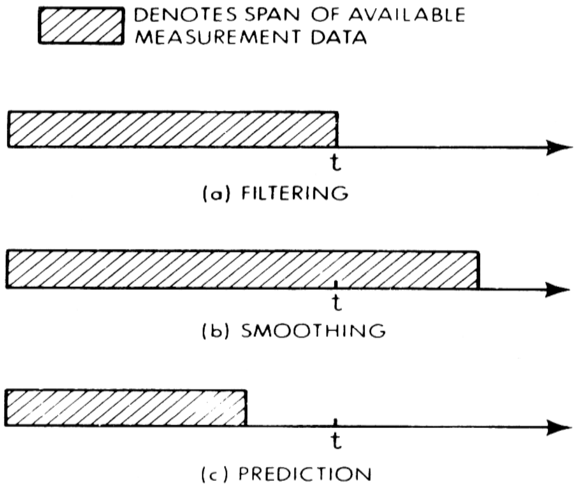
e) A PID controller continuously calculates an *error value* e(t) as the difference between a measured process variable y(t) and a desired set point r(t). The controller attempts to minimize the error over time, by adjustment of a *control variable u(t)*, such as the position of a control valve.

$$u(t) = K_p e(t) + K_i \int_0^t e(t)dt + K_d \frac{de(t)}{dt}$$



P accounts for present values of the error, I accounts for past values of the error, accumulates over time, and *D* accounts for possible future values of the error, based on its current rate of change. Must tune the coefficients Kp, Ki og Kd

f) Filtering uses all past measurements (up to the time tn of interest), while a smoothing makes use of data both before and after any given time point tn of interest.



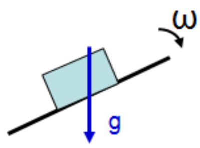
g)

Alpha filter – a first-order approach

- If you have a measurement $\tilde{x}_k$, you can apply a first order filter:

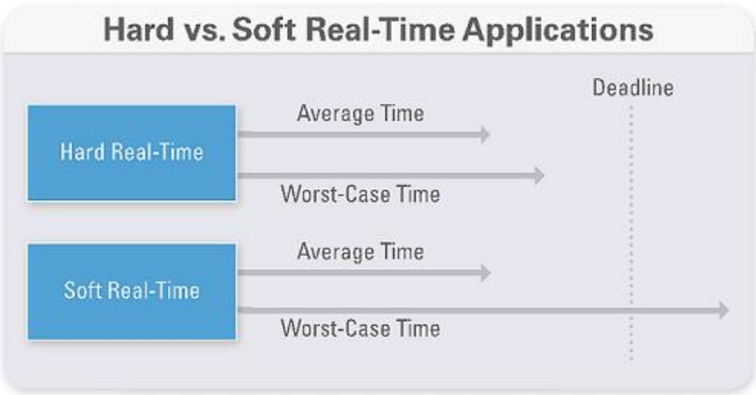
$$\hat{x}_k = (1-\alpha)\bar{x}_k + \alpha\tilde{x}_k$$

- $\hat{x}_k$ is the updated (from measurements) estimate at time k
- $\bar{x}_k$ is the predicted (time propagated) estimate at time k, from a model: $\bar{x}_k = \text{f}(\hat{x}_{k-1})$
  - Data fusion example: rate gyro measurement used for predicting a rotation angle and an accelerometer used as an inclinometer to measure the absolute angle.
- α is a scalar gain between 0 and 1 (typically constant)
- If no measurements $\tilde{x}_k$ are available, α is set to 0 → only prediction
- This approach will filter out noise, but a good α must be found from "trial and error" (possibly with some "guidelines")
- Not as good as a Kalman filter!
  - Not an optimal solution!



Problem 2

a) Explain determinism! Hard and soft real-time (Deadline, average time, worst-case time).

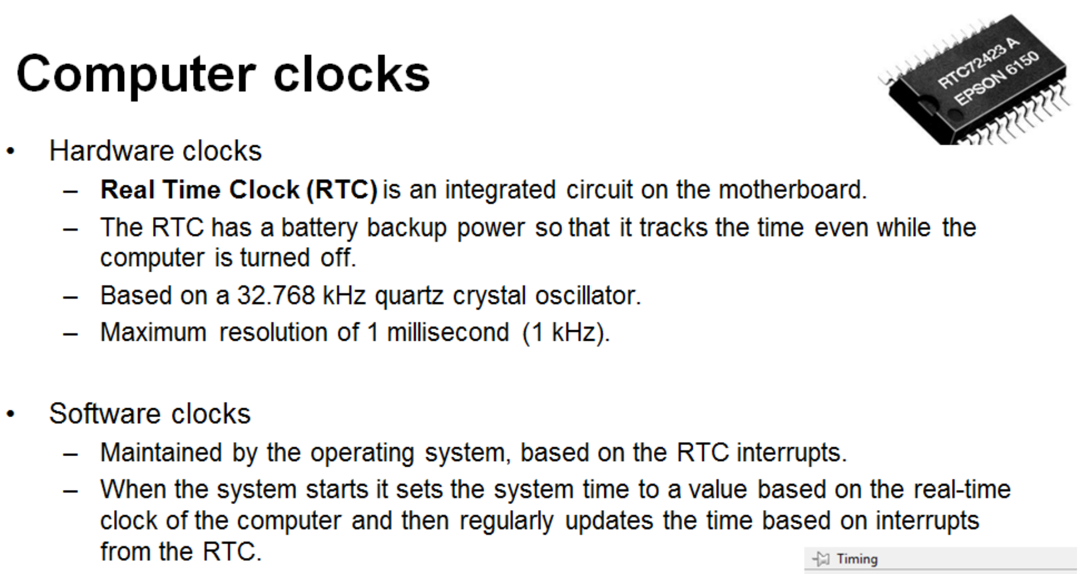


b) DAQ-systems. Buffers used in different parts of the system (FIFO buffer on DAQ-card, PC-buffer/Circular buffer in PC RAM, and queues between different loops/threads).
c) A device driver (from national Instruments), to communicate with DAQ equipment.
d) Not continuous sampling e.g. due to insufficient use of memory (FIFO and circular buffer to compensate for non-real time behavior of the PC), or due to a busy CPU (DMA not used).
e) High data bandwidth and high power/current.
f) For an ideal ADC we can assume an SNR of about 6*N, where N is the number of bits. For 16 bits this gives a SNR of about 96 dB. This means that the stop band of our filter should be at about – 96 dB at 15 kHz. With a sufficiently high filter order this means that the filter cut-off frequency (at -3 dB) can be set at 5 kHz (or above), to give a filter passband from 0 Hz and up to 5 kHz (or higher).
g) Two different architectures for handling memory storage. A big-endian machine stores the most significant byte first, at the lowest byte address. A little-endian machine stores the least significant byte first (at the lowest byte address).

h) A binary file format from National Instruments that is optimized for high-speed streaming.

Problem 3

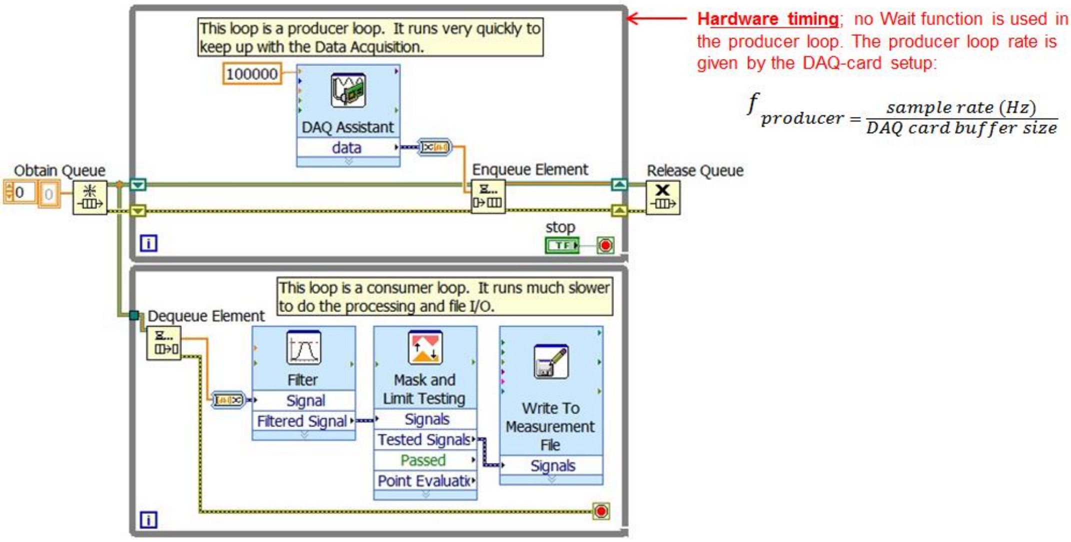
a)



b) Temperature compensation (TCXO) and temperature stabilization (OCXO).

c) Leap seconds added to UTC

d) <u>Hardware timing</u>: <u>A hardware device is in control of the timing</u>; e.g. the sample rate of a DAQ-card is controlled by the onboard DAQ-card oscillator (or a function generator, or a sensor/camera etc.)



<u>Software timing</u>: <u>A wait function is used inside a software loop, using the operating system (software) clock to control the timing of a computer program:</u>