Departament de Disseny i Programació de Sistemes Electrònics
UNIVERSITAT POLITÈCNICA DE CATALUNYA

Embedded Systems
Final exam. June 3, 2016

Duration: 3 hours. Last revision day: June 23

1 Warming up questions

1. Define a real time system (RTS).
2. Situate the course project DTMF in the previous definition of a RTS.
3. What's The difference between the RS-232 and RS-485 serial interfaces?
4. Serial ports have disappeared from PCs but are they really gone?
5. Comment on the paper entitled *Absolutely Positively on Time: What Would It Take?*. If you don't remenber the content of the paper you can focus on the following excerption:

   > Embedded software differs from other software in more fundamental ways. Examining why engineers write embedded software in assembly code or C reveals that efficiency is not their only concern, and may not even be their main one. Reasons for this could include the need to count cycles in a critical inner loop—not to make it fast, but rather to make it predictable.

6. Define polling and interrupt in the embedded systems context. Compare both methods.
7. Focus on the microntroller based solution of the course project DTMF. List in detail the use of polling and/or interrupts. Justify their use. Consider using the other method instead of the one used.
8. The previous question could be made when considering the FPGA based solution? Remember that the FPGA is connected to an Arduino.
9. The previous question could be made when considering the PC or Raspberry Pi based solution?
10. Preliminary results working with the Raspberry Pi based solution, show that the time needed to make the computations involved in the Goertzel algorithm are 100 times lower with a code written in C than with a code written in Python. Comment on theses results.
11. Consider the following situation. A GPS is sending a frame data through an RS232 connection (70 Bytes at 10Hz). A 6-axis IMU must be read each 10ms through an I2C connection. All this data must be saved in an SD memory through an SPI connection. Choose the platform and give as many details as possible about the implementation.

2 C for embedded systems programming

1. Look at the following precedence rules ordered from higher to lower priority.

```
Cast (type)a
Mult a*b a/b
Add a+b a-b
Shift a<<b b>>a
```

Regarding the Goertzel algorithm used in the course project DTMF and using the variables declared below (for the moment forget the promotion rules):

```
int16_t s1=128,s2=4;sn=0;
const int16_t a=8;
uint8_t xn=8;
```

a) Add all the parenthesis you need to make it clear what the following code does. Which is the value of sn?

```
sn=xn+a*s1>>8-s2;
```

b) Add all the parenthesis you need to make it clear what the following code does. Which is the value of sn?

```
sn=a*s1>>8-s2+xn;
```

c) Add all the parenthesis you need to make it clear what the following code does. Which is the value of sn?

```
sn=xn-s2+a*s1>>8;
```

d) Write the right code, minimizing the number of parenthesis, that implements the filter used by the Goertzel algorithm: $sn = xn + ((a * s1)/256) - s2$. Which is the value of sn?

2. Considering promotion rules, explain in detail the size of each product and addition involved in the following code.

```
int32_t X=s1*s1+(int32_t)s1*s1+(int32_t)s2*s2-((int32_t)a*s1>>8)*s2;
```

3 Approximating a real number with a finite number of bits

1. The coefficient that appears in the Goertzel algorithm to detect the power of a signal at 941 Hz, using 205 samples sampled at 8 kHz, has the value $a = 2 * \cos(2 * \pi * \text{round}(941 * 205/8000)/205) \approx 1.4829$. During the course we have scaled a by a factor of 256 and defined the result as a 16-bit signed integer. Scaling this way we have a smaller error: $a_{scaled} = \text{round}(\text{a} * 256) = \text{round}(379.61) = 380$. So, the absolute error after rounding is slightly greater than 0.1%.

   a) Why do we scale by a power of two (i.e 256, 512, 1024...)? Are there powers of two betters than others when using a microcontroller? and when using an FPGA?

   b) Could we obtain a lower error scaling by 512?

   c) By which number should we scale to get an error lower than 0.001%? Could we code a_{scaled} as a 16-bit signed integer?

4 Analog to digital converter

During the project course we have taken samples of a signal that contains a DTMF signaling. When working with Arduino we have used the 10-bit ADC of the AVR microcontroller with a range that goes from ground to an analog reference that can be selected between 1.1 V, 5 V and an external voltage (for example 3.3 V available on the same board). When working with DE0-Nano we have used the 12-bit ADC, external to the FPGA.

1. Consider a DTMF signal that moves between $-0.5\,\mathrm{V}$ and $0.5\,\mathrm{V}$.
   a) Which analog reference will you select when working with Arduino?
   b) Draw a sketch of the circuit used to add an appropriate DC voltage to this signal *prior* to its connection to the ADC input. Which type of filter are we dealing with? Set the values of the components, and compute the cutoff frequency of the filter.

5 Serial communication

1. Samples of 8 bit, sampled at a sampling frequency F_s, are send from an Arduino to the Raspberry Pi (RasPi) through a 57 600 bps serial connection (USART to USB bridge) using 1-start-bit and 1-stop-bit. What is the maximum F_s you could use?

2. Consider the previous Arduino trying to send samples sampled at $F_s = 8\,\mathrm{kHz}$ through a 57 600 bps serial connection. What happens? Quantify the effect.

3. The measured maximum, mean and minimum time used by the RasPi B + to read L bytes, for any L between 1 and 1024, stored in the system buffer are approximately 500 us, 450 us and 300 us respectively. Now consider the following code (in the next questions ignore the execution time of the *print* and the *time.time()* functions).

```
# -*- coding: utf-8 -*-
import time
import numpy
import serial
# parameters
L=...
# open, wait and clean serial port
ser = serial.Serial('/dev/ttyACM0', baudrate , timeout=1)
time.sleep(2)
ser.flushInput()
while 1:
        t0=time.time()
        x=ser.read(L)
        t1=time.time()
        print t1-t0
```

   a) If, as in the previous question, Arduino is sending samples at $F_s = 8\,\mathrm{kHz}$ through a 57 600 bps serial connection to the RasPi, which is the mean of the printed value $t_1 - t_0$? Let $L = 1$.
   b) Now let $L = 100$.

4. Now we add some code to compute something (e.g. an implementation of the Goertzel algorithm) after reading the serial port.

```
while 1:
        t0=time.time()
        x=ser.read(205)
        compute_something(x)
        t1=time.time()
        print t1-t0
```

a) Consider an Arduino sending samples at $F_s = 8\,\text{kHz}$ through a 115 200 bps serial connection to the RasPi. The mean of the printed value $t_1 - t_0 = 25.625\,\text{ms}$. Can you determine the execution time of the *compute_something()* function? Can you limit its value?

b) Repeat the previous question considering that the mean of the printed value $t_1 - t_0 = 27\,\text{ms}$. What percentage of the samples are lost?

c) Repeat the two previous questions changing the baud rate from 115.2 kbps to 250 kbps.

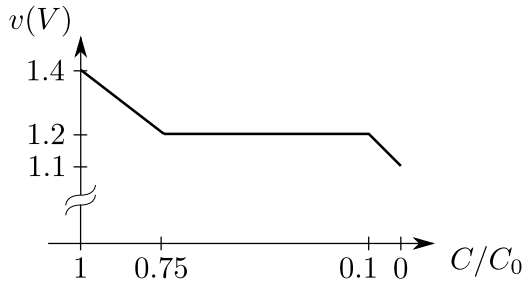
5. If the operating system of the RasPi makes us wait 200 ms, what is the maximum sampling frequency F_s at which samples could arrive without losing them, considering that the size of the system buffer is 4095 bytes? Consider the best case, i.e. when the buffer is empty.

6 Power consumption

Consider that the consumption of a RasPi with some peripherals is 5 W. It is powered by a power supply system made by 2 NiMH AA batteries, each one with a nominal voltage of 1.2 V and a capacity $C_0 = 1800\,\text{mAh}$ (for any current discharge, to make it easy), followed by an appropriate DC-DC converter (with a 90% efficiency).

1. Compute the running time, before the batteries are empty, of the RasPi. Let the voltage battery be the nominal one no matter the percentage discharge.

2. Repeat the previous question considering now that the voltage decreases as the remaining capacity C of the battery decreases as is shown below.



7 Goertzel algorithm

Many of you have chosen the set of parameters $F_s = 8\,\text{kHz}$ and $N = 205$ to, using the Goertzel algorithm, compute the power of a signal at frequencies $F = \{697, 770, 852, 941, 1209, 1336, 1477, 1633\}$. Give as many details as you can of why this set of parameters is a *suitable* one. Are these parameters bounded? Is there any other *suitable* set of parameters? Some of you have used a different value of N. Comment on the benefits of this decision.