Embedded Systems
Final examination. June 18, 2021

Time limit: 150 minutes.

1 Fixed point arithmetic

We want to compute $a * s_1$ in `C` language, where `uint8_t` s_1`=35`. The parameter a must be fixed-point coded with 8 bits.

1. If we code a with $\{B, F\} = \{8, F\}$ (a once coded is named a_k) and we want to approximate $a * s_1$ by an unsigned integer doing `uint8_t y=` $(a_k * s_1)$ `>>` S, what is the value of S to be used?

2. Now consider $a = 1.6$ and $\{B, F\} = \{8, 0\}$. Compute `uint8_t y=` $(a_k * s_1)$ `>>` S. Specify the source of the difference between the approximated value y and the exact value $a * s_1$.

3. Repeat the previous question using now$\{B, F\} = \{8, 1\}$

4. Repeat the previous question using now$\{B, F\} = \{8, 3\}$

2 Fixed point arithmetic

At the end of the second fixed point arithmetic class we talk about how to code the parameter $a_i = 2\cos(2\pi F_i/F_s)$, for the whole range of DTMF frequencies F_i when using the Goertzel algorithm. Knowing that $F_s = 8\,\text{kHz}$, $F_0 = 697\,\text{Hz}$ and $F_7 = 1633\,\text{Hz}$,

1. Determine the minimum a_{min} and maximum a_{max} values of the parameter a.

2. Use $\{B, F\} = \{8, 4\}$ to code a_{min} and a_{max}.

3. Compute the relative error when computing a_{min} and a_{max} with this codification.

4. Compute the relative error of the actual frequencies F_0 and F_7 that are being computed by the Goertzel algorithm.

5. Propose your $\{B, F\}$ values to ensure that the relative error for F_0 is lower that 10%.

3 Embedded system design introduction

1. In the context of of the subject *Embedded Systems* point out the differences between the following terms: embedded systems (ES), cyber-physical systems (CPS), Internet of things (IoT) and industry 4.0 (I4.0).

4 The Goertzel algorithm

The Goertzel algorithm consist in the following two-stage filter:

a) $s_n = x_n + 2\cos(\omega_0)s_{n-1} - s_{n-2}$

b) $y_n = s_n - e^{-j\omega_0}$

The first stage must be computed for each $n = 0...\mathrm{N}-1$ sample and the second stage just the final sample $n = \mathrm{N}-1$.

1. Prove that instead of two-stage filter, the following one-stage filter could be used to obtain the same result.

   c) $y_n = x_n + e^{j\omega_0}y_{n-1}$

2. Compare the computations that must be made for each implementation (two-stage or one-stage) and decide which one to use in a microcontroller like the one find in your Arduino UNO development board.

5 Embedded system design introduction

1. Comment on the following sentence: *Cyber-physical systems must be dependable.*
2. Define *safety*, *security*, *confidentiality*, *reliability*, *repairability* and *availability.*

6 FSM in VHDL

1. Consider a state diagram (or graph) that describes a FSM. Can we say if we are dealing with a Moore or Mealy FSM just looking at the transition arc?
2. Draw the block diagram of a FSM.

7 FSM in VHDL

1. Describe how to implement a FSM using different number of `process` and point out their differences.

8 FPGA and VHDL: synchronizing clock enable with data

A signal called `clk_en_in` is high the first rising edge of `clk` after `data_in` is ready and is low the next rising edge. We want to design an entity in order to update `data_out` from `data_in` every two `clk_en_in`. In addition, a signal called `clk_en_out` must be high the first rising edge of `clk` after `data_out` is ready and must be low the next rising edge. The following `VHDL` code tries to do that.

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;

entity se_exam_2017 is
  port (clk       : in  std_logic;--100kHz
        clk_en_in : in  std_logic;-- 10kHz
        data_in   : in  std_logic;
        clk_en_out: out std_logic;
        data_out  : out std_logic
        );
end entity;

architecture arch_1 of se_exam_2017 is
  signal n : unsigned(7 downto 0):= to_unsigned(0,8);
begin
  process(clk)
  begin
    if rising_edge(clk) then
      if clk_en_in = '1' then
        if n = 1 then
          clk_en_out <='1';
          data_out <= data_in;
          n <= to_unsigned(0,8);
        else
          clk_en_out <='0';
          n <= n+1;
        end if;
      end if;
    end if;
  end process;
end architecture;
```

1. Unfortunately, `clk_en_out` is not well generated. Draw the actual digital waveform (`clk`, `clk_en_in` and `clk_en_out`).

2. Modify the code and draw the right digital waveform.

9 FPGA: hardware reuse

1. What is the strategy behind hardware reuse? Or, if you prefer, what do we lose and what do we gain?

10 C: storage class modifiers

1. Argue why we should use the `const` modifier even if the code is going to work well without it.
2. Can we have a variable with both, `const` and `volatile`, modifiers, like in
   `volatile const a;` ?

11 C: optimization

1. Comment on the main optimizations flags that can be used.
2. Do you think that the -O0 optimization flag has any utility?
3. What happens when many developers are involved in the same project coding different parts of it?
4. Which optimization flag have you used in your DTMF project and why?

12 FPGA: the look up table

1. What is the LUT size of the FPGA (Cyclone IV EP4CE22F17C6N) that you have used?
2. Describe an experiment to verify this size.

13 FPGA: soft and hard multipliers

1. How many hard multipliers do you think will be used to perform and 16x*16 bits multiplication using 9x9 bits hard multipliers.
2. After experiencing with the DE0-Nano board do you have something to say?

14 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 $F_s = 8\,\text{kHz}$ 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 $F_s = 8\,\text{kHz}$ through a 57 600 bps serial connection to the RasPi, what 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 100 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.

15 Contact bounce

We presented some *debouncing* solutions to the contact bounce problem inherent to switches.

1. Point out the different advantages and disadvantages between hardware and software solutions without describing any of them.

2. Some of these solutions needed to be tuned for each contact while others, the so called *universals* solutions, not. Enumerate the *universals* solutions and describe one of them in detail.