Politecnico di Torino

Integrazione di Sistemi Embedded

DSO DESIGN

DIGITAL STORAGE OSCILLOSCOPE ON MCU-BASED BOARD

Bononi Andrea (287628)

CONTENTS

1 SPECIFICATIONS **2**

1.1 Communication Protocol 2

2 DESIGN **3**

2.1 Infinite Loop 3

2.2 Command Identification and System Settings Modification 4

2.3 Block Elaboration and Transmission 4

2.4 End of Transmission 6

2.5 Reception 6

2.6 End of Conversion 7

2.7 Graphical User Interface 8

SPECIFICATIONS

- MCU-based board: STM32F401RE
- Input channels: 2
- Input range: fixed gain, 0 V - 3.3 V unipolar
- Digitizer resolution: 8 bit - 256 levels
- Sampling frequency: 10 Hz - 100 kHz
- Trigger type: Auto/Normal/Single/Stop
- Trigger level: DC coupled, 0 V - 3.3 V
- Trigger position: ± 128 samples
- Display resolution: 256 × 256 points

The overall system is made up by two different sections:

- **Acquisition system** realized on an MCU-based board. It samples the analog signals, digitalize them (based on the trigger conditions) and send the acquired data to a PC by means of a standard RS232 full-duplex serial line. The acquisition parameters can be set from the PC through the same RS232 interface used to send the acquired data.
- **Graphical user interface** running on a standard PC. It is implemented in Python and it must be able to display the acquired data on the screen and to configure the acquisition board.

1.1 Communication Protocol

The protocol used on the RS232 channel is ASCII based. Any character not included between the start character (*) and the stop character (#) must be silently discarded.

The following **configuration commands** can be sent from the PC to the acquisition board:

- ***SPxxxxxxxx#** → set the sampling frequency of input channels. The parameter *xxxxxxxx* is an 8 digit hexadecimal number, representing the required sampling period in nanoseconds.
- ***TLxx#** → specify the trigger level used to synchronize the time base to the input signal. The trigger event is generated when the input signal of channel 1 crosses the selected trigger level with a positive (rising) slope. The parameter *xx* is a 2 digit hexadecimal number, representing the required trigger level.
- ***TTxx#** → select one of four different synchronization modes (trigger types) according to the parameter xx. Possible values are:
  - **Automatic** (00): the trigger event is generated if the trigger condition is satisfied or if a delay greater then an entire horizontal sweep time has elapsed.
  - **Normal** (01): the trigger event is generated only if the trigger condition is satisfied.
  - **Single** (10): the trigger event is generated as in normal mode, but for one time only (then the acquisition subsystem is stopped).
  - **Stopped** (11): The acquisition is stopped, no trigger events are generated.

The acquisition results are sent from the acquisition system to the PC in the format ***aabbaabbaabb.....#**. After a trigger event, when the acquisition subsystem has acquired enough samples (internal buffer full) it sends back data to the PC in this format. The samples of the two channels are interleaved and sent as hexadecimal two-digits numbers. In this notation, *aa* is the sample of the first input channel, while *bb* is the corresponding sample of the second input channel. The number of sample pairs is always greater than 256 and the central pair represents the data acquired when the trigger event occurred.

DESIGN

Data are sent from the acquisition board to the PC in blocks of 511 samples (255 samples before the trigger event and other 255 samples after the trigger event). Considering both the channels, a total of 1022 samples are included between the start character and the stop character and sent to the PC. Each transmission corresponds to a LED blinking, so that we can have a visual feedback about the transmission speed. If any error occours, the LED remains turned on.

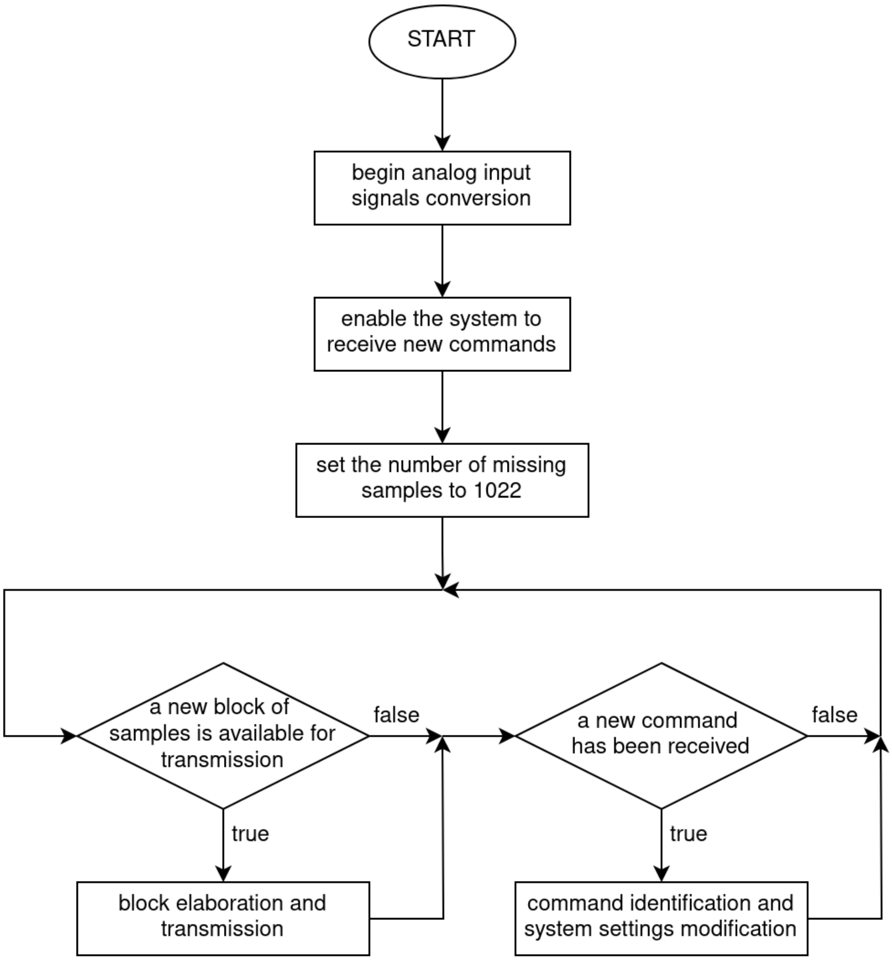
The main pheripherals employed in the design are listed below:

- **ADC**: it samples the analog input signals and digitalizes them
- **UART**: it manages the communication with the PC
- **TIMER**: it controls the timing of the sampling system in order to achieve the desired sampling frequency

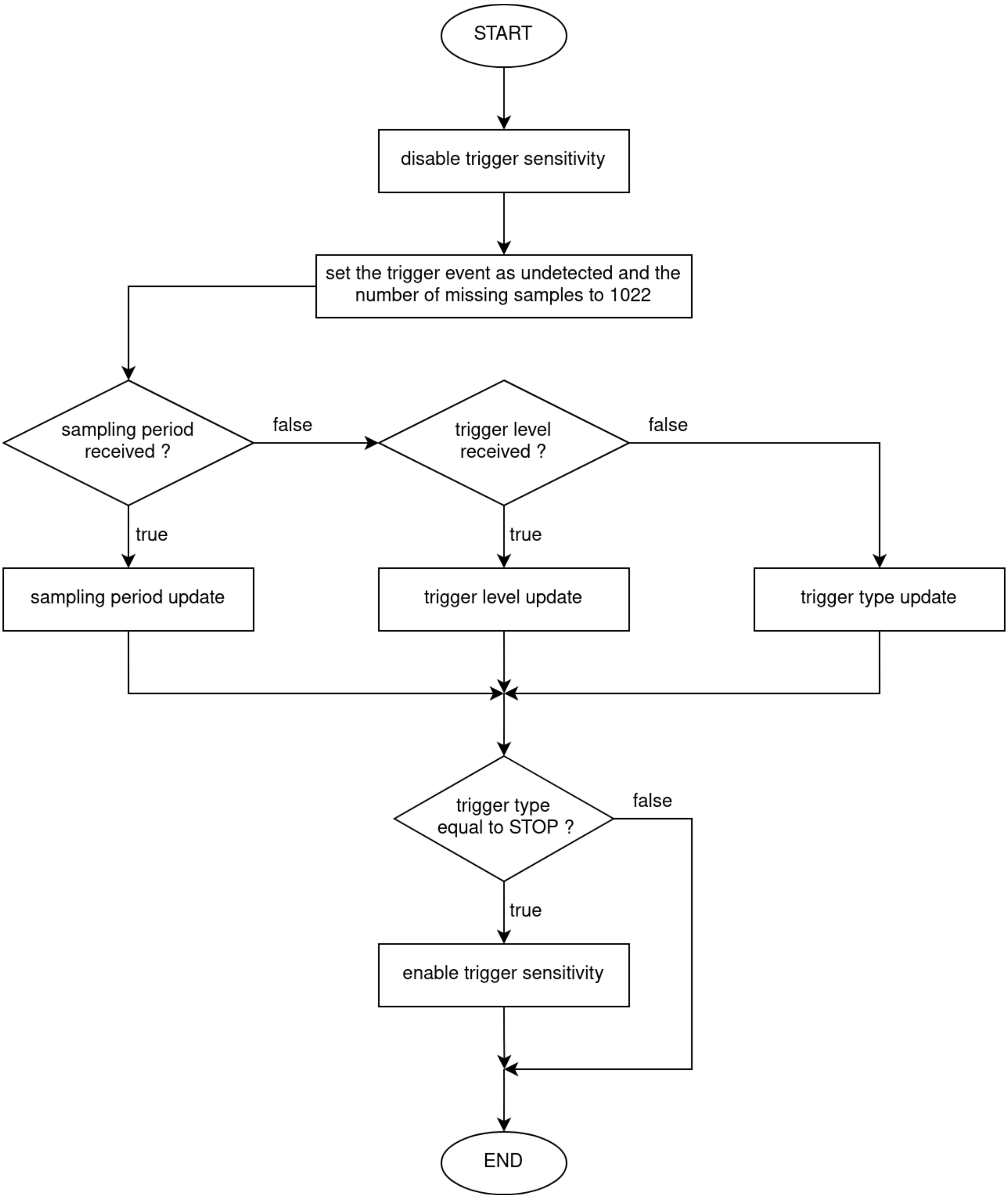
All the peripherals work at the same frequency, i.e. 84 MHz. The timer prescaler is set to 168, therefore its reload value must be in the range 5 - 50000 to guarantee the desired sampling frequency range. The analog-digital converter uses a regular channel (CH0) and an injected channel (CH1) working in auto-injected mode. The two channels are sampled one after the other and the results are stored in two different registers in order to prevent possible overruns.

2.1 Infinite Loop

The infinte loop looks for a new block of data to be transmitted or a new received command. It works in polling.


START
begin analog input signals conversion
enable the system to receive new commands
set the number of missing samples to 1022
a new block of samples is available for transmission
false
true
block elaboration and transmission
a new command has been received
false
true
command identification and system settings modification

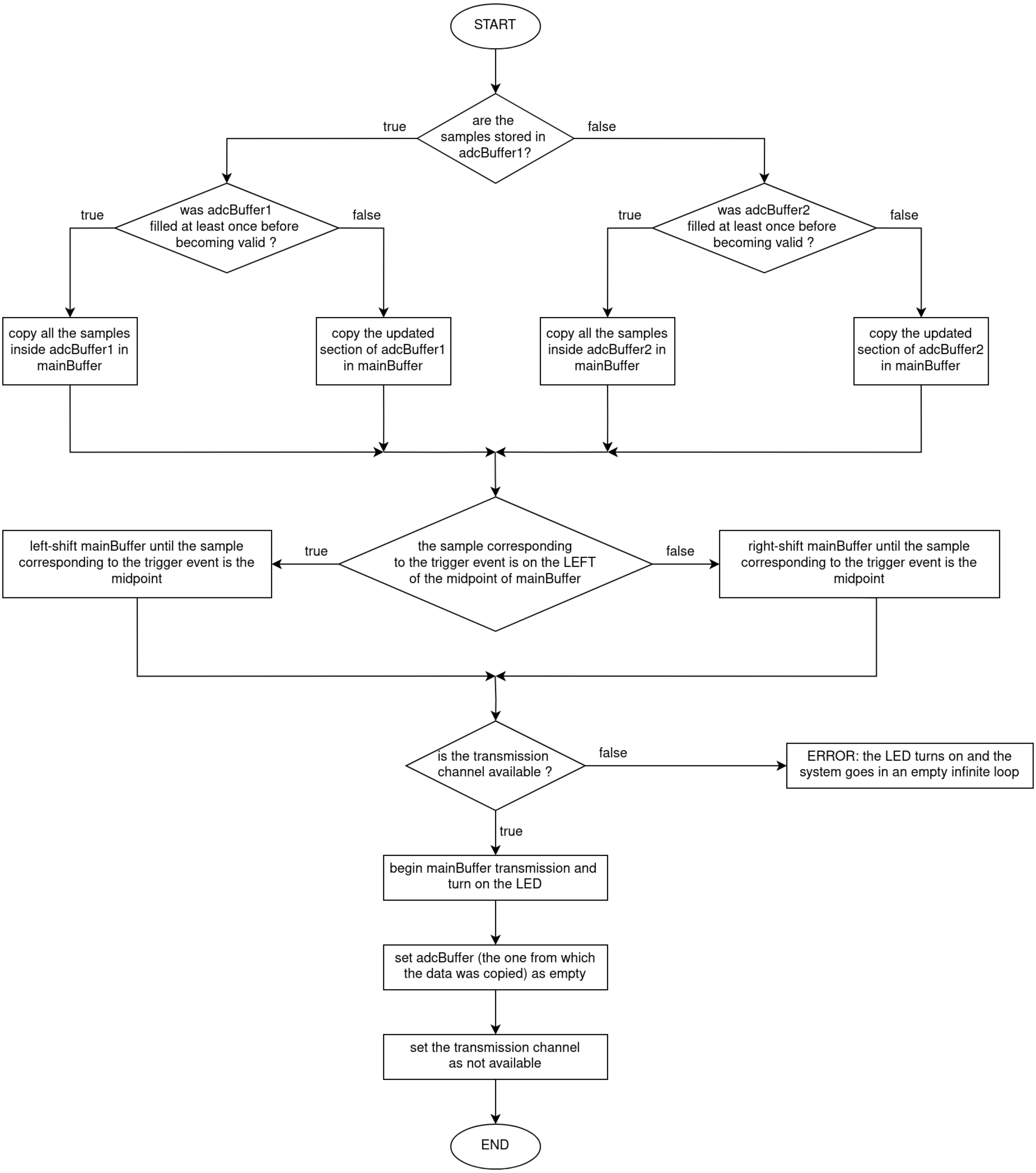
2.2 Command Identification and System Settings Modification


START
disable trigger sensitivity
set the trigger event as undetected and the number of missing samples to 1022
sampling period received ?
false
trigger level received ?
false
true
true
sampling period update
trigger level update
trigger type update
trigger type equal to STOP ?
false
true
enable trigger sensitivity
END

2.3 Block Elaboration and Transmission

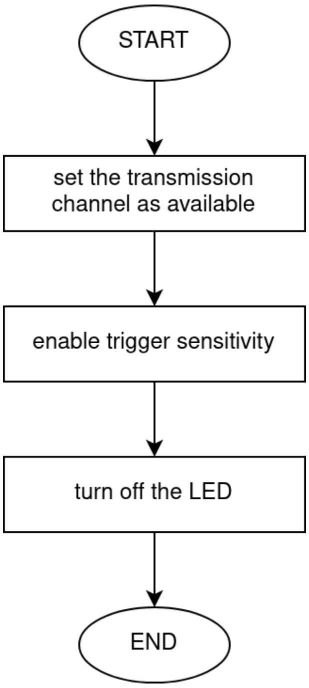
The system employs two different buffers (adcBuffer1 and adcBuffer2) to store the samples. When a certain buffer contains a valid data block (which must be elaborated and transmitted), the new samples are stored in the other buffer and viceversa. The valid data block is copied in a third buffer (mainBuffer), in which it can be elaborated before being transmitted.

The following flowchart describes all the steps required to begin the transmission of mainBuffer. The end of the transmission is described in the next section.

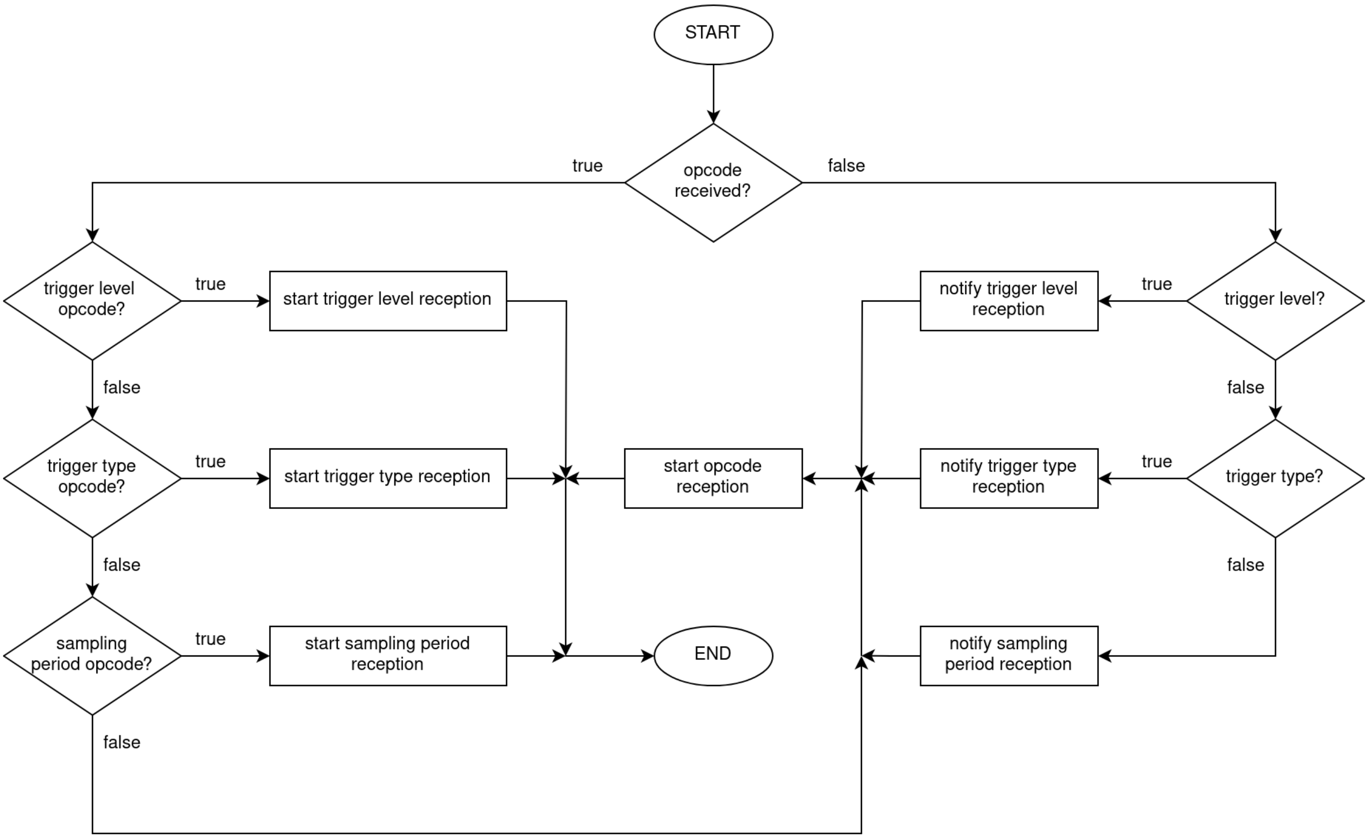

START
are the samples stored in adcBuffer1?
true
false
was adcBuffer1 filled at least once before becoming valid ?
true
false
was adcBuffer2 filled at least once before becoming valid ?
true
false
copy all the samples inside adcBuffer1 in mainBuffer
copy the updated section of adcBuffer1 in mainBuffer
copy all the samples inside adcBuffer2 in mainBuffer
copy the updated section of adcBuffer2 in mainBuffer
left-shift mainBuffer until the sample corresponding to the trigger event is the midpoint
true
the sample corresponding to the trigger event is on the LEFT of the midpoint of mainBuffer
false
right-shift mainBuffer until the sample corresponding to the trigger event is the midpoint
is the transmission channel available ?
false
ERROR: the LED turns on and the system goes in an empty infinite loop
true
begin mainBuffer transmission and turn on the LED
set adcBuffer (the one from which the data was copied) as empty
set the transmission channel as not available
END

Considering a very high working frequency, it is possible that during the elaboration and transmission of a valid data block inside a certain adcBuffer, the other adcBuffer is filled multiple times (it is circular buffer). On the other hand, working at very low frequencies it can reach the validity without being filled even once.

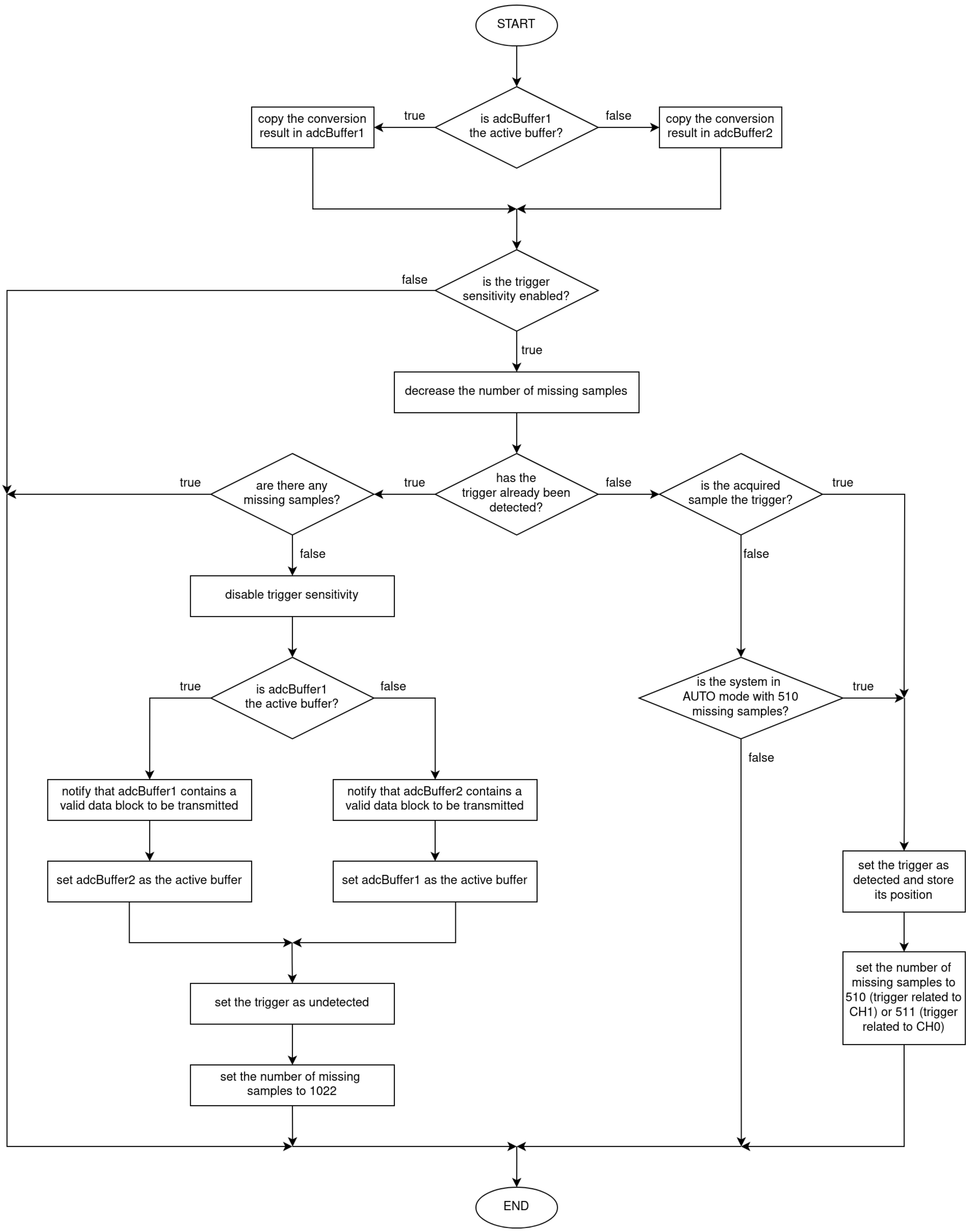
2.4 End of Transmission


START
set the transmission channel as available
enable trigger sensitivity
turn off the LED
END

2.5 Reception



2.6 End of Conversion


START
is adcBuffer1 the active buffer?
true
false
copy the conversion result in adcBuffer1
copy the conversion result in adcBuffer2
is the trigger sensitivity enabled?
false
true
decrease the number of missing samples
has the trigger already been detected?
true
false
are there any missing samples?
true
false
is the acquired sample the trigger?
true
false
disable trigger sensitivity
is adcBuffer1 the active buffer?
true
false
notify that adcBuffer1 contains a valid data block to be transmitted
notify that adcBuffer2 contains a valid data block to be transmitted
set adcBuffer2 as the active buffer
set adcBuffer1 as the active buffer
set the trigger as undetected
set the number of missing samples to 1022
is the system in AUTO mode with 510 missing samples?
true
false
set the trigger as detected and store its position
set the number of missing samples to 510 (trigger related to CH1) or 511 (trigger related to CH0)
END

2.7 Graphical User Interface

The GUI has been realized in Phyton using the Tkinter module and the Matplotlib module, together with the pyserial module to manage the serial communication with the board.

Other than plotting the acquired samples, the GUI allows the user to modify the sampling parameters (trigger level, trigger value, trigger position and sampling period). If non-valid values are inserted, the previous values are restored after showing an error message (nothing is transmitted to the board). As far as the sampling period is concerned only integer values are allowed, although it is possible to modify the measurement unit (s, ms, us, ns).

After connecting the board to your PC, it is possible that during the first tun of the GUI an RX/TX conflict arises, leading the board in its error state (LED constantly turned on). In this case, exit the GUI and reset the board using the dedicated pushbutton and restart the GUI.