Communication Protocol for Testing Impedance Microbiology

Synshine Engineering Team

Using the circuit provided, we will be scanning multiple frequencies, starting from the frequency value closest to 30 Hz, and every subsequent frequency will be two times the previous frequency. The circuit will be tested for 14 different frequencies.

The following things are black-boxed:

- int init_adc() : Initializes ADC. If it worked, then returns 0, else -1.
- int init_sine() : Initializes the sine wave generator.If it worked, then returns 0, else -1.
- long read_val_adc(int a) : This function does the following :
  - If 'a' is any one of 0,1,2, or 3, it returns the 24-bit recorded by a^{th} channel of the ADC as a long under normal operation.
  - Otherwise -1
- int set_val(long freq) : This sets the closest frequency possible in the sine wave generator.If it worked, then return 0, else -1.
- int change_freq_steps(long n_steps) : Adds *n_steps* to the 32-bit integer which is given as an input to the sine wave generator. Basically, if *n_steps* is positive, it sets the sine wave generator to the n_steps^{th} smallest frequency among all the frequencies generatable by the sine wave generator which are higher than the one its currently set to. Returns 0 in case of normal operation, -1 in case of overflow, -2 in case of other errors.
- int mult_freq(long k) : Mutiplies the 32-bit integer, which is given as an input to the sine wave generator, with k, similar to the previous function. Returns 0 in case of normal operation, -1 in case of overflow, and -2 in case of other errors.
- int set_sine(bool a) : Turn the sine wave generator on if a is true, and turn it off otherwise. Returns 0 in case of normal operation, -1 otherwise.
- int set_gain(int a): Set the gain of the adc. Returns 0 in case of normal operation, -1 in case the value was not suitable, -2 in case of other errors.

The arduino responds to the command of the computer through serial communication.

1 Transmission Protocol for the Primary message

The primary message will be the initial message before validation procedures.

1.1 From Computer to Arduino

The computer can send the following (# is 32-bit integer, represented as 4 consecutive chars):

- SETFR:# = This commands the arduino to set the frequency to the value closest to # which is a long
- CGSTP:# = This commands the arduino to change the frequency steps by # which is a long
- MLSTP:# = This commands the arduino to multiply the frequency by # which is a long
- CHKCF: = This commands the arduino to check the current frequency
- GENHI: = This commands the arduino to turn on the sine wave generator

- GENLO: = This commands the arduino to turn off the sine wave generator
- ERROR:<> = This sends an error to the arduino
- SETGN:# = This commands the arduino to change the gain of the adc to #

1.2 From arduino to computer

The arduino can send the following:

- ERROR:<> = This sends an error to the computer
- SDDAT:#:c = This sends the data to the computer as # = ADC output of the c^{th} ADC

2 Verification Protocol

The next communication will be a set of verification bits.

2.1 Transmission of input message

A message packet will be of the form "ST<MESSAGE>ab\r\n". Here,

- ST signifies the start of one message packet
- <MESSAGE> is the placeholder for the message sent.
- 'a' is a char which is a byte-wise XOR of the characters in the message.
- 'b' is (Sum of integer representations of char is MESSAGE) mod 2^8
- "\r\n" signifies the end of one message packet.

2.2 Receiving of the message

On recieving the message, the following is performed :

1. If the first two chars are ST, continue. Else send ERROR
2. Compare 'a' with the 'a' that is computed using the message. If same, continue.Else send 'ERROR' back to the sender
3. Compare 'b' with the 'b' that is computed using the message. If same, continue.Else send ERROR back to the sender.
4. Check is the last two chars. If they are \r\n, continue.Else send ERROR back to the sender.
5. Send 'CNFa' back to the sender if the message is confirmed, where 'a' is the same as in previous section.
6. In case ERROR was sent back, the serial buffer is emptied.

2.3 Confirmation of transmission

The sender keeps resending the message packet till a 'CNFa' is recieved. There is a gap of 10 mS to ensure the reciever has had enough time to process a message packet.