TÉCNICO LISBOA

Instrumentação e Aquisição de Dados

1º Trabalho

Aquisição de dados e comunicação

Licenciatura Bolonha em Engenharia Física Tecnológica (LEFT)

Instituto Superior Técnico, Universidade de Lisboa

Francisco Carreira, ist196527

Francisco Mendes, ist196529

Joana Abreu, ist196235

Mariana Ribeiro, ist196552

Grupo nº 3, IAD

Este trabalho tem como intuito implementar um sistema em que se liga por USB um Raspberry PI e um Arduino.

Com este trabalho pretende-se desenvolver um programa $Python\ 3$, de forma a que o Raspberry PI envie, em intervalos bem definidos (no nosso programa escolhemos intervalos de 50 ms), um comando ao Arduino para que este adquira o valor analógico associado à voltagem de um pin deste (no nosso caso o $pin\ A0$) e o envie de volta. Os valores recebidos do Arduino serão, posteriormente, colocados num $plot$ em janela QT, onde existem também os botões $Start$ e $Stop$ e uma caixa de texto onde o utilizador pode enviar um comando para o Arduino para imprimir o valor analógico adquirido. No caso do programa que desenvolvemos, o comando válido a enviar para o Arduino para aquisição do valor é "0", pelo que qualquer outro carácter introduzido resultará na impressão de "COMANDO INVALIDO".

Rasberry PI

O código desenvolvido para o bom funcionamento do Raspberry PI encontra-se no ficheiro $trabalho1.py$. Começámos por definir a janela, após importarmos as devidas bibliotecas. Posteriormente, definimos o gráfico e os botões $Start$ e $Stop$. Para além disto, foi criada uma caixa onde o utilizador pode introduzir um comando, "$textbox$", associada a um botão que envia o texto para o Arduino.

A função $fArduino$ trata de enviar o que é escrito na "$textbox$"para o Arduino. Esta função é ativada de duas formas possíveis: carregando na tecla "Enter"quando se está na "$textbox$", ou carregando no botão $Enviar$. As função $fStart$ e $fStop$ dão a ordem de início e de paragem ao cronómetro (que diz ao programa quando deve realizar a atualização do $plot$) após o utilizador carregar nos botões $Start$ e $Stop$, respetivamente. Aqui, reparou-se que ao clicar duas ou mais vezes seguidas no botão $Start$ o cronómetro começava de novo, acontecendo um pequeno lag ("soluços"); para resolver esta situação, impusemos uma condição para que a função $fStart$ só dê início ao cronómetro se este ainda não estiver ativo. E, finalmente, definiu-se a função que trata da atualização do $plot$, $update\_plot$. Dentro desta função, o buffer é limpo e é escrito o comando que permite a leitura do valor de voltagem no pin (comando para o Arduino "0"), depois o código para durante 0.03s para assegurar que a leitura pelo Arduino e a escrita do valor de voltagem no buffer acontecem sem sobreposições. Após recomeçar, o programa testa o valor recebido para garantir que este é um float e, caso seja, atualiza a lista de pontos e consequentemente o gráfico.

Arduino

Por outro lado, o código desenvolvido para o Arduino, que se encontra no ficheiro $trabalho1.ino$ começa por definir as variáveis utilizadas, passando, de seguida, para a definição de duas funções necessárias para o bom funcionamento do Arduino: $setup$ e $loop$, ambas do tipo $void$.

Dentro da função $setup$, começámos por definir a taxa de dados em bits por segundo para transmissão serial de dados com o valor 9600. Posteriormente, definimos o $pin\ A0$ como $input$. Já dentro da função $loop$, após verificar que há algo escrito no $buffer$ enviado pelo Raspberry PI, é lido o valor recebido no $buffer$. De seguida, verifica-se se o comando enviado pelo Raspberry PI é válido, isto é, se é igual a "0". Caso o comando seja válido, é realizada a leitura do valor no $pin\ A0$, que corresponde a um número entre 0 e 1023, caso contrário, o programa imprime "COMANDO INVALIDO".

Para obtermos o valor da voltagem, realizámos uma normalização do valor lido, dividindo-o por 1023, multiplicando-o também por 5, uma vez que se trata do valor máximo da voltagem. Por fim, o valor é escrito no $buffer$ para posterior leitura pelo Raspberry PI.

Foi possível implementar todas as características do sistema propostas com sucesso e não se encontram erros na sua utilização após variados testes ao programa.