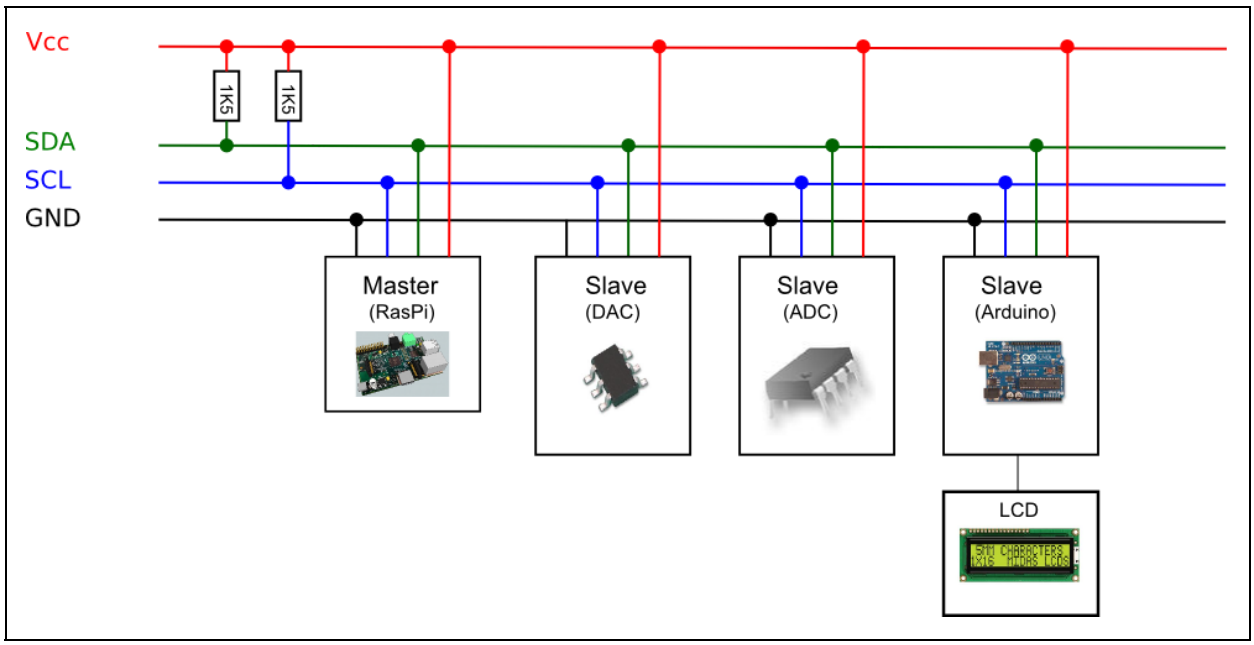
Introdução

Um robô frequentemente usa mais Arduinos do que o número de portas USB disponíveis em seu computador central. Conectar vários Arduinos à mesma porta usando um hub USB se mostrou uma solução pouco confiável. Como solução, a equipe decidiu usar o protocolo I²C para interligar Arduinos.

O protócolo I²C permite conectar dispositivos com apenas 2 fios (assumindo que seus terras já estejam conectados). É um barramento com topologia mestre-escravo e vários dispositivos (IMU, LCD, sensor de temperatura, …) suportam interface por I²C. A figura abaixo ilustra uma configuração típica:



Para aprender mais recomenda-se este tutorial da SparkFun: http://sfe.io/t82

No Arduino, a biblioteca Wire.h permite usar o I²C. No entanto, devido às seguintes características decidiu-se implementar uma biblioteca auxiliar (WireBotz.h):

- As funções da Wire.h são baixo nível, trabalhando a nível de byte.
- Não há conferência de erro.
- O tamanho de uma menssagem é limitado a 32 bytes.

Na versão 1.0, a biblioteca melhora o primeiro ponto.

Instalação

A biblioteca está na pasta WireBotz, no repositório git de módulos da equipe. As instruções de instalação, também presentes no readme.txt são:

```
INSTALANDO

Para usar a biblioteca:
      1. Encontrar o diretorio de bibliotecas da sua IDE do Arduino. No
Linux costuma ser ~/sketchbook/libraries .
      2. Copiar a pasta ./copy_to_libraries/WireBotz para o diretorio de
bibliotecas.
      3. Após reiniciar a IDE do Arduino, o menu " Sketch > Import Library
" deverá conter o item WireBotz
      4. Para usar a biblioteca no seu código, é necessário usar os dois
includes:
            #include <Wire.h>
            #include <WireBotzXxxx.h>
         Onde Xxxx é Master ou Slave, dependendo do papel do Arduino.
```

Exemplos

Há uma pasta “exemplos”. Nela estão os códigos para uma montagem com um mestre e um escravo. A montagem está explicada nos códigos:

```
//  VISAO GERAL e MONTAGEM
//
//  Estes programas implementam o exemplo clássico de um LED com o brilho
//  controlado por potenciômetro. A inovação é que um arduino (slave) lê o
//  valor do potenciômetro (0 a 1024) que é enviado ao outro arduino (master)
//  No master o valor é convertido para um inteiro entre 0 e 255 e enviado de
//  volta ao escravo, que finalmente ajusta o brilho do LED
//                          ____________________              ____________________
//    Potenciometro ====>|                    |     I2C      |                    |
//                       | Arduino (slave) |<------------>| Arduino (master)  |
//         LED <=========|____________________|              |____________________|
//
//  O codigo assume os seguintes pinos na montagem:
//  Slave:
//     3: conectado ao anodo do LED
//    A0: conectado ao pino central do potenciomentro
//    A4: I²C SDA (linha de dados)
//    A5: I²C SDC (linha de clock)
//  Master:
//    A4: I²C SDA (linha de dados)
//    A5: I²C SDC (linha de clock)
//
//  Dizer que os pinos A4 do slave e do master estão conectados na linha SDA
//  significa que os pinos A4 do arduino serão conectados um ao outro. Além
//  disso um resistor de pull up (na faixa 1k ~ 10k) deve ser conectado a cada
//  uma das linhas (SDA e SCL).
```

Lista de Funções

Master:

- **Master::begin(byte address)**
  Inicializa o Arduino como mestre
- **Master::write(byte addr, uint8_t msg[], int size)**
  Envia dados em msg para o escravo de endereço addr
- **Master::read(byte addr, uint8_t msg[], int maxSize)**
  Lê dados do escravo de endereço addr e coloca em msg

Slave:

- **Slave::begin(byte address)**
  Inicializa o Arduino como escravo
- **Slave::setTXBuffer(uint8_t buffPtr[], int size)**
  Registra o Buffer de envio
- **Slave::setRXBuffer(uint8_t buffPtr[], int size)**
  Registra o Buffer de recebimento
- **Slave::newMessage()**
  Retorna verdadeiro se o escravo recebeu uma nova menssagem do mestre.
  NB: Após retornar verdadeiro uma vez a função retorna falso até a próxima menssagem chegar.

A biblioteca já declara os objetos Master e Slave. Então é possível usar o comando Master.begin() sem declarar um objeto, da mesma forma que a biblioteca Serial.h já tem o objeto Serial disponível só de incluir a biblioteca.

As funções estão documentadas em maior detalhe nos arquivos WireBotzMaster.h e WireBotzSlave.h . O exemplo incluído com a biblioteca explica bem como integrar as funções em seu programa.