VII Jornada de **Iniciação Científica**, **Desenvolvimento Tecnológico** e **Inovação** do Ifes

INSTITUTO FEDERAL ESPÍRITO SANTO

DESENVOLVIMENTO DE INTERFACE DE COMUNICAÇÃO E EXPANSÃO PARA KIT DE ROBÓTICA EDUCACIONAL

Daniel Barcelos Mendes, Felipe Nascimento Martins

Programa Institucional de Bolsas de Iniciação Científica - PIBIC
Campus Serra
Instituto Federal do Espírito Santo - Ifes

d_bm3@hotmail.com, felipemartins@ifes.edu.br

Resumo - O presente trabalho faz parte de um projeto que pretende implementar a Robótica Educacional como ferramenta de trabalho pedagógico interdisciplinar para cursos profissionalizantes de nível técnico e superior. Consiste na criação de uma interface de comunicação entre o kit de robótica da LEGO (NXT) e a plataforma Arduino. Esta plataforma foi escolhida devido à sua praticidade e amplo leque de sensores compatíveis. Foram realizados estudos referentes tanto à montagem quanto à programação dos dois controladores (NXT e Arduino) e, em seguida, montados alguns protótipos para testes dos programas iniciais. O protocolo adotado para o desenvolvimento da comunicação entre o NXT e o Arduino foi o I²C devido à sua simplicidade e à existência de bibliotecas em ambas as linguagens, que facilitam a implementação do sistema. Após a obtenção do entendimento necessário de ambas as linguagens deu-se início à construção e programação da comunicação propriamente dita. Para prova de comunicação foi utilizado um sensor de luz tipo LDR (resistor dependente de luz) e implementado o seguinte algoritmo: caso estivesse muito escuro o Arduino enviaria um sinal para o LEGO e este acionaria o seu sensor emissor de luz. Outros testes também foram realizados utilizando-se outros dispositivos como um sensor de distância ultrassônico e um servomotor. Os resultados dos testes foram satisfatórios e demonstraram que o sistema implementado cumpriu com sucesso os objetivos propostos no plano de trabalho original.

Palavras-chave: LEGO, Arduino, I²C, robótica.

Abstract - This work is part of a project that aims to implement Educational Robotics as an interdisciplinary pedagogic tool for professionalizing courses (technical high-school level). It consists in the creation of a communication interface between the Lego NXT robotics kit and the Arduino platform. This platform was chosen because of its convenience and wide range of compatible sensors. Studies have been conducted about the assembly and the programming of both controllers (NXT Arduino) and then some prototypes were built for testing purposes. The I²C protocol was adopted to implement the communication between the NXT and the Arduino due to its simplicity and availability of libraries in both languages, which facilitate the implementation of the system. A test-bed was implemented to test the communication. It was used a light sensor type LDR (light dependent resistor) and implemented the following algorithm: if it was too dark, the Arduino would send a signal to the LEGO and this one would turn on its light emitting sensor. Other tests were also conducted using other devices as an ultrasonic distance sensor and a servomotor. The test results were satisfactory and demonstrated that the implemented system successfully met the objectives proposed in the original work plan.

Key-words: LEGO, Arduino, I²C, robotic.

INTRODUÇÃO

A utilização de robôs tem grande importância em áreas como fabricação de automóveis, exploração de petróleo em águas profundas, monitoramento de tubulações (como oleodutos), trabalho em setores potencialmente perigosos (como na manipulação de elementos radioativos), monitoramento aéreo autônomo de grandes áreas (para busca, vigilância,

verificação de desmatamento, monitoramento de áreas cultivadas etc.), todas atividades de grande importância para o Brasil. Além dessas áreas, a robótica também tem aplicação na área educacional. Segundo Santos e Menezes [1], a Robótica Educacional pode ser definida como "um ambiente onde o aprendiz tenha acesso a computadores, componentes eletromecânicos (motores, engrenagens, sensores, rodas etc.), eletrônicos (Interface de *hardware*) e um ambiente de programação para que os componentes acima possam funcionar".

A Robótica Educacional é considerada uma área essencialmente interdisciplinar [2]. Em um ambiente pedagógico composto por dispositivos robóticos há o constante diálogo de diversas disciplinas como a matemática, a física, a psicologia, a medicina, a computação, dentre outras. Para d'Abreu [3], ao elaborar-se um ambiente de aprendizagem baseado no uso de dispositivos robóticos, pode-se dar tanto o enfoque técnico-industrial quanto o pedagógico-educacional.

A utilização da robótica como ferramenta educacional se mostra interessante por dois fatores principais. Primeiro, por promover um ambiente de incentivo para que os alunos se mantenham motivados a se dedicarem a atividades e carreiras ligadas à tecnologia. Segundo, por possibilitar ao aluno o contato e a utilização de tecnologias estudadas em várias disciplinas de seu curso, permitindo a aplicação de conceitos na solução de problemas reais.

Nesse contexto, os kits de robótica que foram adquiridos pelo IFES-Serra são muito interessantes didaticamente, por permitirem rápida implementação de diferentes robôs, com diferentes estruturas mecânicas, sensores e tipos de acionamento. No entanto, possuem limitações em termos de número de sensores que podem ser utilizados simultaneamente, capacidade de armazenamento e comunicação de dados etc. O sistema aqui proposto pretende ampliar tais capacidades possibilitando a construção e estudo de robôs com maior número de sensores e a implementação de algoritmos de controle mais complexos.

METODOLOGIA

Para atingir o objetivo final do trabalho, que era construir uma interface de comunicação entre o kit da LEGO e a plataforma microcontrolada Arduino, foram desenvolvidas algumas etapas a serem seguidas passo a passo. Primeiramente houve um período de intensas pesquisas sobre o funcionamento dos dois microcontroladores onde se obteve o maior número de informações necessárias para o desenvolvimento do projeto. Os sites oficiais de ambos os controladores serviram como uma das principais fontes de pesquisa por disponibilizarem informações confiáveis, inclusive os "*datasheets*" de cada um.

Com as informações em mãos foi dado início à fase de estudos, após a qual foram realizados diversos testes para colocar em prática os conceitos estudados. A partir destes testes (realizados separadamente com cada microcontrolador) pôde-se ter um conhecimento maior do amplo leque de funções e bibliotecas presentes em cada linguagem e também dos periféricos de cada kit, ou seja, sensores e atuadores. Em seguida definiu-se qual o protocolo de comunicação a ser utilizado, levando em consideração principalmente a praticidade de trabalho.

De posse dos conhecimentos das linguagens, da parte física e também do protocolo de comunicação a ser utilizado, construiu-se então um protótipo para teste de comunicação e foram desenvolvidos os códigos de comunicação tanto na interface do LEGO como na

interface do Arduino. Após o sistema pronto, iniciou-se a etapa de testes finais onde foram feitas as devidas correções e ajustes.

Desta forma, a execução total do projeto, desde a fase de estudos até a fase de testes pode ser dividida em várias atividades a seguir definidas:

- *Estudo do funcionamento do kit de robótica LEGO NXT:*
Foram pesquisadas diversas fontes as quais dizem respeito às características físicas construtivas e às características de programação do kit da LEGO. As buscas se concentraram na internet devido à vasta quantidade de sites [4] [5] [6] [7] que tratam do assunto tendo maior destaque o próprio site da LEGO Mindstorms [6] foi tirado um grande número de informações relevantes para o projeto. Também foram pesquisados materiais a respeito da IDE BricxCC, interface escolhida para programação do kit [4]. Desta forma, a internet foi a maior fonte de pesquisa utilizada para realização deste trabalho.

- *Escolha do microcontrolador a ser utilizado:*
Antes da execução do projeto já existia uma ideia de usar o Arduino (Figura 1) como plataforma microcontrolada. Esta ideia foi confirmada depois de um estudo mais detalhado do material. A escolha do Arduino se deve principalmente à praticidade de trabalho que a plataforma apresenta. O microcontrolador já vem embutido na placa que também já apresenta memória para armazenamento do programa, memória para armazenamento de dados volátil e não volátil, facilidade de comunicação com o PC para upload do programa (via USB), além do baixo custo que fica em torno de oitenta reais [9]. Outro ponto positivo da placa é a grande quantidade de informação disponível na internet que dispõe inclusive de um site oficial [9] com informações sobre a plataforma.

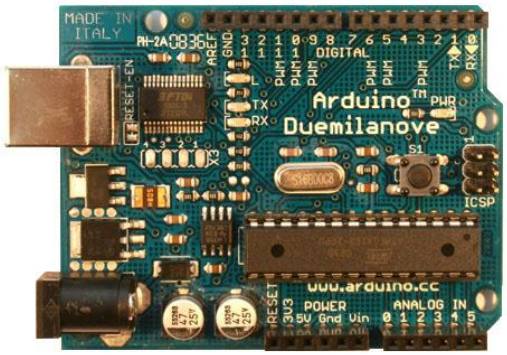

Figura 1. Plataforma Arduino

- *Estudo do microcontrolador escolhido (Arduino):*
Foram realizados vários estudos sobre a plataforma microcontrolada Arduino com o intuito de aprofundar mais os conhecimentos acerca de sua programação e seus componentes. O site oficial do Arduino [9] dispõe de todas as informações necessárias para quem deseja iniciar algum trabalho com a plataforma. O estudo compreendeu a parte de *hardware* (conhecimento da pinagem, alimentação etc.) e a parte de *software* (funções, bibliotecas, detalhes da linguagem etc.).

- *Construção de protótipos para testes individuais com o Arduino e com o LEGO:*
Após a obtenção dos conhecimentos necessários para iniciar a execução do projeto, deu-se início à parte de construção de protótipos para testes individuais. Essa etapa consiste na

criação de alguns modelos simples utilizados para colocar em prática o que foi estudado na etapa anterior. Os primeiros protótipos eram individuais, ou seja, não havia nenhuma parte do código voltada para comunicação. Como exemplo de teste, para o Arduino pode-se citar o "identificador de senha" e para o LEGO o "robô empilhadeira" (Figura 2), cujo código será mostrado no item a seguir.

Figura 2. Robô Empilhadeira

- *Desenvolvimento dos programas para os testes individuais:*
Com os protótipos construídos e já com a ideia do que os mesmos deveriam fazer, deu-se início à etapa de elaboração das programações de teste para cada protótipo. Alguns códigos foram desenvolvidos, e abaixo segue o exemplo supracitado, o "robô empilhadeira":

```
#include "NXCDefs.h"
task main()
{
 SetSensorType(S4, SENSOR_TYPE_LOWSPEED);
 OnFwd(OUT_BC, 50);
 Wait(2000);
 Off(OUT_BC);
 Wait(500);
 OnRev(OUT_A, 20);
 Wait(3200);
 Off(OUT_A);
 Wait(500);
 OnFwd(OUT_BC, 50);
 while (SensorUS(IN_4)>18);
 Off(OUT_BC);
 Wait(500);
 OnRev(OUT_A, 20);
 Wait(8000);
 Off(OUT_A);
 Wait(500);
 OnFwd(OUT_BC, 30);
 Wait(1300);
 Off(OUT_BC);
 Wait(500);
 OnFwd(OUT_A, 20);
 Wait(1100);
 Off(OUT_A);
 Wait(500);
 OnRev(OUT_BC, 50);
 SetSensorType(S3, SENSOR_TYPE_COLORRED);
 Wait(2000);
```

```
 OnFwd(OUT_B, 50);
 Wait(1500);
 OnFwd(OUT_BC, 50);
 Wait(2000);
 Off(OUT_BC);
}
```

- *Definição do protocolo de comunicação a ser utilizado;*
De todos os protocolos de comunicação existentes, dois chamaram mais atenção e se mostraram mais eficientes para este projeto: o serial (RS485) [10] e o I²C. Ambos apresentam compatibilidade com as duas plataformas adotadas, ou seja, tanto o microcontrolador do LEGO como o do Arduino possuem funções e bibliotecas desenvolvidas especificamente para esses protocolos. No entanto, o escolhido como ferramenta para este projeto foi o protocolo I²C devido ao grande número de funções já preparadas especialmente para ele em ambos os controladores e também devido ao maior número de informações disponíveis no site do Arduino [9] e do LEGO [6], quando comparado ao RS485.

- *Estudo do protocolo de comunicação escolhido (I²C);*
Após a definição do protocolo, iniciou-se o estudo mais aprofundado do mesmo. O barramento I²C consiste fisicamente de dois fios e uma conexão com o terra. Os dois fios são bidirecionais, SDA *(Serial Data Line*) e SCL (*Serial Clock Line*). Os dispositivos são interconectados através destes dois sinais, formando um barramento I²C. Neste barramento, um dispositivo pode atuar como MASTER ou SLAVE, sendo que o MASTER é o responsável pela comunicação [11]. No caso do LEGO e do Arduino, o primeiro atua como mestre (MASTER) "requisitando" as informações necessárias do Arduino e este, por sua vez, atua como escravo (SLAVE).

- *Construção do protótipo e desenvolvimento do programa com a comunicação entre as duas interfaces (LEGO e Arduino);*
Depois de realizados todos os estudos e testes individuais e definido o protocolo, a comunicação passou a ser desenvolvida. Primeiro pensou-se em um protótipo para servir como base oficial do projeto. Este protótipo consistia em um robô que se movimentaria em direções aleatórias desviando dos obstáculos. O Arduino foi então interligado a este robô (LEGO) e em um *protoboard* montou-se um circuito com um LDR, de forma que sempre que a luz ambiente diminuísse significativamente o sensor enviaria um sinal para o LEGO via I²C e este acionaria seu sensor emissor de luz.

- *Testes finais;*
Etapa de ajustes e testes. Nesta foram identificados os erros e então corrigidos. Foi a última etapa da implementação do projeto. Os testes consistiram em uma série de procedimentos executados um a cada vez com o intuito de identificar os possíveis erros no código de programação do LEGO e do Arduino. Erros de sintaxe e lógica são comuns quando se trabalha com a linguagem C/C++. Para identificação desses erros utilizou-se a estratégia de ir "debugando" o código de programação linha por linha, ou seja, linhas de código foram introduzidas em partes estratégicas do corpo principal do programa devendo ser mostradas no display do LEGO sempre que tais partes fossem executadas. Desta forma foi possível identificar e corrigir os locais onde ocorriam erros, como por exemplo, travamento do programa.

RESULTADOS

A comunicação entre o LEGO e o Arduino foi obtida de forma a atender o objetivo do projeto. A velocidade de comunicação se mostrou suficiente para atender aos objetivos, ou seja, permite uma comunicação de informação sensorial em tempo hábil para que o robô possa tomar uma ação de controle em resposta ao estímulo do sensor. O protocolo de comunicação I²C atendeu bem as expectativas devido ao fato de ser relativamente simples e de já existirem bibliotecas em ambas as linguagens com algumas funções que ajudam na comunicação. A foto do protótipo final (Figura 3) e o código final de programação do Arduino e do LEGO estão apresentados a seguir.

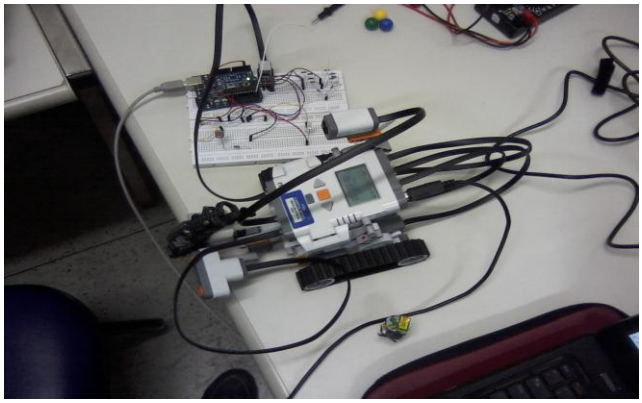

Figura 3. Protótipo para teste de comunicação

CÓDIGO DO LEGO:

```
//PROGRAMA PARA TESTE DE COMUNICAÇÃO DO ARDUINO COM O LEGO NXT

/*
OBJETIVOS:
- O robô deve se movimentar em direções aleatórias.
- Quando encontrar algum obstáculo deve atirar e em seguida desviar do mesmo.
- Se o ambiente estiver escuro (informação enviada pelo arduino) o robô deve acionar o sensor emissor de luz.

INFORMAÇÕES:
- Endereço do Arduino visto pelo Lego: 0x43 << 1
- Registro: é o byte de informação enviado pelo Lego ao Arduino
- Arduino conectado à porta IN_1 do Lego
*/


// ------------------------INÍCIO DO CÓDIGO---------------------------------------- //

#include "NXCDefs.h"

// -------------------BLOCO DECLARAÇÃO DE VARIÁVEIS---------------------- //

int aux, aux1, aux2, aux3;
string light;
byte ARDUINO_ADRESS_CONTACT =  0x43 << 1;  //endereço do arduino visto pelo Lego


// -------------------BLOCO FUNÇÃO DE COMUNICAÇÃO---------------------- //

string i2cReceptData(byte port, byte adr, byte reg, byte count)
{
           //TextOut(0, LCD_LINE4, "TESTE_DEBUG4");
           string data;
           byte receiveData[];
           byte sendData[];
           ArrayBuild(sendData, adr, reg);
           while (I2CCheckStatus(port) ==  STAT_COMM_PENDING)
           {}
           //TextOut(0, LCD_LINE4, "TESTE_DEBUG5");
           Wait (100);
```

```
           if(I2CBytes(port, sendData, count, receiveData))
           {
                           data = ByteArrayToStr(receiveData);
                           //TextOut(0, LCD_LINE4, data);
           }

           return data;

 }


// -------------------BLOCO MOVIMENTO DO LEGO---------------------- //

task move()
{
 //TextOut(0, LCD_LINE2, "TESTE_DEBU7");
 while(true)    //Rotina responsável por fazer o robô andar em direções aleatórias
 {
   until (aux==0);
   aux=1;
   OnFwd(OUT_AC, 50);
   aux=0;
   Wait(Random(4000));
   until (aux==0);
   aux=1;
   OnRev(OUT_C, 50);
   aux=0;
   Wait(Random(3300));
   Off(OUT_AC);
 }
}


// -------------------BLOCO ATIRAR BOLINHAS---------------------- //

task shoot()
{
 //TextOut(0, LCD_LINE4, "TESTE_DEBU12");
 while(true)     //Rotina responsável por fazer o robô detectar um obstáculo e atirar
 {
  if(SensorUS(IN_4) <= 15)
  {
   until (aux==0);
   aux=1;
   Off(OUT_AC);
   Wait(1000);
   OnFwd(OUT_B, 60);
   Wait(750);
   Off(OUT_B);
   Wait(1000);
   OnFwd(OUT_A, 50);
   Wait(Random(1000));
   aux=0;
  }
 }
}


// -------------------BLOCO DE COMUNICAÇÃO---------------------- //

task communication()
{
     //TextOut(0, LCD_LINE4, "TESTE_DEBU13");
     while(true)    //lê a informação passada pelo arduino, ou seja, se está escuro ou não
     {
          Wait (100);
          //TextOut(0, LCD_LINE4, "TESTE_DEBUG2");
          light = i2cReceptData(IN_1, ARDUINO_ADRESS_CONTACT, 1, 1);
          //TextOut(0, LCD_LINE4, light);
          if (light == "A" )
          {
               //TextOut(0, LCD_LINE4, "TESTE_DEBUG3");
               SetSensorType(S3, SENSOR_TYPE_COLORRED);
               TextOut(0, LCD_LINE4, "AMBIENTE ESCURO");
          }
          else
          {
```

```
                //TextOut(0, LCD_LINE4, "TESTE_DEBUG3");
                SetSensorType(S3, SENSOR_TYPE_COLORNONE);
                TextOut(0, LCD_LINE4, "AMBIENTE CLARO");
           }
      }
}


// -------------------BLOCO DE INICIALIZAÇÃO--------------------- //

task main()      //rotina principal
{
     aux = 0;
     Precedes(communication, move, shoot);
     SetSensorType(IN_4, SENSOR_TYPE_LOWSPEED);
     SetSensorMode(IN_4, SENSOR_MODE_RAW);
     Wait(1000);
     SetSensorLowspeed(IN_1);
     //TextOut(0, LCD_LINE4, "TESTE_DEBUG1");
}
```

CÓDIGO DO ARDUINO:

```
//PROGRAMA PARA TESTE DE COMUNICAÇÃO DO ARDUINO COM O LEGO NXT
/*
OBJETIVOS:
- O robô deve se movimentar em direções aleatórias.
- Quando encontrar algum obstáculo deve atirar e em seguida desviar do mesmo.
- Se o ambiente estiver escuro (informação enviada pelo arduino) o robô deve acionar o sensor
emissor de luz.

INFORMAÇÕES:
- Endereço do Arduino: 0x43
- Endereço do Arduino visto pelo Lego: 0x43 << 1
- Registro: é o byte de informação enviado pelo Lego ao Arduino
- Arduino conectado à porta IN_1 do Lego
*/
// ----------------------------INÍCIO DO CÓDIGO---------------------------------- //

#include <Wire.h>

// -------------------BLOCO DECLARAÇÃO DE VARIÁVEIS---------------------- //

int LED = 2;
int LDR = A0;
byte auxiliar;


// -------------------BLOCO DE INICIALIZAÇÃO--------------------- //

void setup()
{
  pinMode(LED, OUTPUT);
  pinMode(LDR, INPUT);
  Serial.begin(9600);
  Wire.begin(0x43);
  Wire.onRequest(respondeLEGO);
  Wire.onReceive(recebeLEGO);
}


// ---------BLOCO FUNÇÃO EXECUTADA QUANDO O ARDUINO RECEBE ALGUM DADO DO LEGO------------ //

void recebeLEGO(int howMany)  //FUNÇÃO OK
{
  if(Wire.available() > 0)
  {
    auxiliar = Wire.read();
    //Serial.println(auxiliar);
}
}


// -------------------BLOCO FUNÇÃO EXECUTADA QUANDO O LEGO SOLICITA ALGUM DADO DO ARDUINO-----
---------------- //
```

```
void respondeLEGO()
{
  byte luz;
  if(auxiliar == 1)
  {
   //Serial.println("ok");
   if (analogRead(LDR) <= 408) //Ambiente muito escuro
   {
      //Serial.println("ok");
      luz = 0x41; // Representa a letra "A"
      digitalWrite(LED, HIGH); // Acende led
      Wire.write(luz); //Envia dado para Lego
 }
   else //Ambiente claro
 {
      Serial.println("ok");
      luz = 0x42;  // Representa a letra "B"
      digitalWrite(LED, LOW); //Apaga led
      Wire.write(luz); //Envia dado para o Lego
 }
 }
}


void loop()
{

  delay(100);
}
```

DISCUSSÃO E CONCLUSÕES

Com a interface básica da comunicação construída se torna possível a implementação de códigos com quaisquer tipos de sensores aceitos pelo Arduino, visto que cada sensor tem a sua especificidade mas o código da comunicação não precisa ser alterado significativamente de acordo com cada sensor. Porém, alguns problemas surgiram durante a fase de testes e devem ser levados em consideração em um possível trabalho futuro:

- Diferença de bits no endereço: O endereçamento do Arduino é de oito bits enquanto o do LEGO é de apenas sete. Para resolver este problema foi necessário dar um *leftshift* no endereço do Arduino dentro do código do LEGO;
- Registro: O protocolo de comunicação I²C exige o fornecimento de um *byte* de registro que funciona como um endereço de onde será lida a informação em questão. Porém, para requisitar um dado do Arduino não foi preciso fornecer o *byte* de registro por parte do LEGO, pois o Arduino possui apenas um local de onde as informações são lidas;
- Resistores de *pull-up*: A plataforma Arduino já possui resistores de *pull-up* internos não havendo a necessidade de acrescentar mais nenhum externamente. Já o LEGO NXT não possui esses resistores internos e então foi necessário acrescentar dois resistores de 82k (segundo *datasheet* do LEGO) externamente. Sem estes resistores a comunicação não funcionou.

AGRADECIMENTOS

Ao professor Dr. Felipe Nascimento Martins.
Ao Instituto Federal do Espírito Santo.

REFERÊNCIAS

[1] SANTOS, C. F.; MENEZES, C. S. A Aprendizagem da Física no Ensino Fundamental em um Ambiente de Robótica Educacional. Anais do XI Workshop de Informática na Escola, do XXV Congresso da Sociedade Brasileira de Computação, São Leopoldo, RS, 2005.

[2] FRANSICO JÚNIOR, N. M.; VASQUES, C. K.; FRANSISCO, T. H. A. Robótica Educacional e a Produção Científica na Base de Dados da CAPES. Revista Electrónica de Investigación y Docencia, n. 4, p. 35-53, 2010.

[3] D'ABREU, J. V. V. Desenvolvimento de Ambientes de Aprendizagem Baseados no Uso de Dispositivos Robóticos. Anais do X Simpósio Brasileiro de Informática na Educação – SBIE99. Curitiba, PR, 1999 .

[4] BRICXCC. Site oficial. Disponível em: <http://bricxcc.sourceforge.net/>. Acesso em: Novembro de 2011 e outros.

[5] BENEDETTELLI, D. Programming LEGO NXT Robot's using NXC. Versão 2.2. 2007.

[6] LEGO. Site oficial. Disponível em: <http://mindstorms.lego.com/en-us/Default.aspx>. Acesso em: Novembro de 2011 e outros.

[7] HANSEN, J. Not Exactly C (NXC) Programmers's Guide. Disponível em: <http://pt.scribd.com/doc/70790219/40/Types-and-Modes>. Acesso em: Outubro de 2011 e outros.

[8] SANTOS, P. N. Arduino: Introdução e Recursos Avançados. 2009.

[9] ARDUINO TEAM (site oficial do Arduino). <http://arduino.cc/en/>. Acesso em: Outubro de 2011 e outros.

[10] ROSTRICH's BLOG. NXT and Arduino talking RS-485. Disponível em: <http://rostrich.wordpress.com/2010/05/16/nxt-and-arduino-talking-rs-485/>. Acesso em: Outubro de 2011 e outros.

[11] PRADO, S. Barramento I²C. 2007. Disponível em: <http://www.embarcados.com.br/index2.php?option=com_content&do_pdf=1&id=83>. Acesso em: Novembro de 2011 e outros.