**RELATÓRIO DE ATIVIDADES PRÁTICAS EM LABORATÓRIO**

**APRESENTAÇÃO DE PROJETOS**

**Curso: Engenharia da Computação**

**Termo: 10° Termo**

**Disciplina: Projeto de Sistemas Embarcados II**

**Projeto: Comunicação para controle de motores.**

**Acadêmico(s):**

|  |  |  |
| --- | --- | --- |
| **RA** | **Nome** | **Assinatura** |
| **207180** | **Lucas Fernando Costa dos Santos** |  |
| **207540** | **Pedro César Vagner Nogueira** |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

***Professor(es):***

**Amadeu Zanon Neto**

**Araçatuba-SP**

**2020**

**Centro Universitário Católico Salesiano Auxilium**



Araçatuba / SP

**RELATÓRIO DE ATIVIDADES PRÁTICAS EM LABORATÓRIO – APRESENTAÇÃO DE PROJETOS**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **CURSO** | **TERMO** | **PERIODO** | **DISCIPLINA** | |
| **ENGENHARIA DA COMPUTAÇÃO** | **10** | **NOTURNO** | **PROJETOS DE SISTEMAS EMBARCADOS II** | |
| **PROFESSOR(A)** | | | **LABORATÓRIO / SALA** | **ATIVIDADE** |
| **AMADEU ZANON NETO** | | | **EAD** | **02/2020** |

**Aluno(s):**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **#** | **RA** | **Nome** | **#** | **RA** | **Nome** |
| **1** | 207180 | Lucas Fernando Costa dos Santos | **6** |  |  |
| **2** | 207540 | Pedro César Vagner Nogueira | **7** |  |  |
| **3** |  |  | **8** |  |  |
| **4** |  |  | **9** |  |  |
| **5** |  |  | **10** |  |  |

**Função ou tarefa atribuída a cada aluno da equipe:**

|  |  |  |  |
| --- | --- | --- | --- |
| **#** | **Função/atribuição** | **#** | **Função/atribuição** |
| **1** | Circuito | **6** |  |
| **2** | Programação | **7** |  |
| **3** |  | **8** |  |
| **4** |  | **9** |  |
| **5** |  | **10** |  |

**Projeto ou atividade a ser desenvolvida – Resumo, enunciado ou descrição: Exercício ( X ) - Trabalho ( ) - Avaliação ( )**

|  |
| --- |
| Neste projeto iremos fazer a integração de três sistemas de comunicação, RS-232, I²C e SPI.  O sistema deverá controlar a rotação de três motores de 12V através de potenciômetros digitais MCP 41010 determinando-se a tensão de saída neles para controle por transístores.  Um supervisório criado em Visual Basic deverá controlar estes motores através da comunicação serial e que deverá passar por um PIC 16F887 que mostrará o percentual de 0 a 99 em displays de sete segmentos para os três motores. Este controle deverá ser feito por multiplexadores PCF8574 e comunicação I²C. Este comando é passado para um PIC 16F690 que através de comunicação SPI controlará os potenciômetros digitais que enviarão a tensão à base dos transístores e assim controlando a rotação dos motores. |

**Desenhos, esquemáticos ou croqui de modelo proposto para o trabalho ou projeto:**

|  |
| --- |
|  |

**Datas e prazos:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Etapa** | **Prevista** | | **Efetiva** | |
| Apresentação do projeto pelo professor: | Data: **28/08/2020** | Hora: **19:05** | Data: **28/08/2020** | Hora: **19:05** |
| Início do projeto: | Data: **28/08/2020** | Hora: **19:05** | Data: **28/08/2020** | Hora: **19:05** |
| Encerramento, entrega ou apresentação: | Data: **18/09/2020** | Hora: **19:05** | Data: \_\_\_\_/\_\_\_\_/\_\_\_\_ | Hora: \_\_\_:\_\_\_ |

**Forma de apresentação:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Prevista / Solicitada** | **Efetiva** | **Nota, ciência e avaliação do professor** | |
| ( ) Montagem prática | ( ) Montagem prática | **Professor: Amadeu Zanon Neto** | |
| (X ) Seminário / Apresentação / Banner | ( ) Seminário / Apresentação / Banner | **Nota/Pontuação** | Assinatura/Visto/Carimbo |
| (X ) Relatório | ( ) Relatório |
| (X ) Simulação | ( ) Simulação |
| ( ) Visita | ( ) Visita |
| ( X ) Mostra / Evento | ( ) Mostra / Evento |
| ( ) Artigo | ( ) Artigo | **Data** | |
| ( X ) Desenvolvimento de software | ( ) Desenvolvimento de software |
| ( ) Monografia | ( ) Monografia |
| ( ) Outro \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | ( ) Outro \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |

**Observações/anotações:**

|  |
| --- |
|  |
|  |
|  |
|  |
|  |
|  |

**Componentes e materiais utilizados:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **#** | **Qte** | **Descrição** | **#** | **Qte** | **Descrição** |
| **1** |  | ARDUINO | **21** |  |  |
| **2** | 2 | MICROCONTROLADOR PIC | **22** |  |  |
| **3** |  | PROTOBOARD | **23** |  |  |
| **4** | 9 | RESISTOR 220R | **24** |  |  |
| **5** | 2 | RESISTOR 10K | **25** |  |  |
| **6** | 3 | LED | **26** |  |  |
| **7** |  | CHAVE TÁCTIL | **27** |  |  |
| **8** | 6 | DISPLAY 7 SEGMENTOS | **28** |  |  |
| **9** |  | DISPLAY LCD | **29** |  |  |
| **10** | 1 | COMPIM | **30** |  |  |
| **11** | 6 | 4511 | **31** |  |  |
| **12** | 3 | MOTOR DC | **32** |  |  |
| **13** | 3 | POTENCIÔMETRO DIGITAL 41010 | **33** |  |  |
| **14** | 2 | CRYSTAL | **34** |  |  |
| **15** | 4 | CAPACITOR | **35** |  |  |
| **16** | 3 | IRF720 | **36** |  |  |
| **17** | 3 | DIODO 1N4007 | **37** |  |  |
| **18** |  |  | **38** |  |  |
| **19** |  |  | **39** |  |  |
| **20** |  |  | **40** |  |  |

**Equipamentos e materiais de laboratório:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **#** | **Qte** | **Descrição** | **#** | **Qte** | **Descrição** |
| **1** | 2 | COMPUTADOR PC | **13** |  | FIOS |
| **2** |  | FONTE DE ALIMENTAÇÃO | **14** |  | SHIELD ARDUINO |
| **3** |  | CAIXA DE FERRAMENTAS | **15** |  |  |
| **4** |  | MULTÍMETRO | **16** |  |  |
| **5** |  | GRAVADOR DE MICROCONTROLADOR | **17** |  |  |
| **6** |  | CLP – MODELO  | **18** |  |  |
| **7** |  | SENSORES INDUSTRIAIS | **19** |  |  |
| **8** |  | DISPLAY DE LCD | **20** |  |  |
| **9** |  | IMPRESSORA 3D | **21** |  |  |
| **10** |  | KIT ROBÓTICO LEGO | **22** |  |  |
| **11** |  | KIT ROBÓTICO TAMIYA | **23** |  |  |
| **12** |  | ESTANHO | **24** |  |  |

**Softwares utilizados**

|  |  |  |  |
| --- | --- | --- | --- |
| **#** | **Descrição /versão** | **#** | **Descrição/Versão** |
| **1** | **MikroBasic free** | **6** |  |
| **2** | **Proteus 7** | **7** |  |
| **3** | **Visual Studio 10** | **8** |  |
| **4** | **Virtual Serial Ports Emulator** | **9** |  |
| **5** |  | **10** |  |

**Conclusões e finalização de projeto**

**Problemas ou dificuldades encontradas e suas soluções (Se necessário anexar folhas suplementares):**

|  |  |
| --- | --- |
| **Problemas e ou dificuldades** | **Soluções propostas e aplicadas.** |
| Dificuldade na lógica de envio de valores. | Foi utilizado valores do tipo *double*. |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**Conclusões e comentários finais**

|  |
| --- |
| Com este projeto foi possível ver a utilização da comunicação I2C e SPI trabalhando com conjunto com a integração de hardware |
| e software via controle remoto, utilizando comunicação RS-232 serial. E com isso foi possível compreender melhor sobre essas |
| tecnologias de comunicação. |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |

**ANEXO - I**

**Programação (Pode ser substituída por listagem impressa).**

|  |
| --- |
| program motores  dim recebe as string[6]  dim valor as integer  main:  TRISA=%11111111  TRISB=%00111111  TRISC=%00000000  INTCON=%00000000  ANSEL=%00000000  ANSELH=%00000000  portA=0  portB=0  portC=0  USART\_INIT(9600)  portc.0 = 1 portc.1 = 1 portc.2 = 1  SPI\_INIT  clearbit(portc,0) Spi\_Write(17) Spi\_Write(0) setbit(portc,0)  clearbit(portc,1) Spi\_Write(17) Spi\_Write(0) setbit(portc,1)  clearbit(portc,2) Spi\_Write(17) Spi\_Write(0) setbit(portc,2)  executa:  if Usart\_Data\_Ready() > 0 then  Usart\_Read\_Text(recebe,"/")  valor = StrToInt(recebe)    if (valor > 999) and (valor < 4256) then    if (valor >= 1000) And (valor < 1256) then  clearbit(portc,0)  valor = valor - 1000  end if  if (valor >= 2000) And (valor < 2256) then  clearbit(portc,1)  valor = valor - 2000  end if  if (valor >= 3000) And (valor < 3256) then  clearbit(portc,2)  valor = valor - 3000  end if  if (valor >= 4000) And (valor < 4256) then  portc.0 = 0  portc.1 = 0  portc.2 = 0  valor = valor - 4000  end if  spi\_write(17)  spi\_Write(valor)  portc.0 = 1  portc.1 = 1  portc.2 = 1  delay\_ms(10)    end if    Usart\_Write\_Text(recebe)    end if  goto executa  end. |
| program displays  dim recebe as string[6]  dim valorRecebido as integer  dim valorPorcent as integer  dim uni as integer  dim dez as integer  dim valorPortas as integer  dim escrita as integer  main:  TRISA=%11111000  TRISB=%11111111  TRISC=%10000001  INTCON=%00000000  ANSEL=%00000000  ANSELH=%00000000  portA=0  portB=0  USART\_INIT(9600)  I2C\_INIT(100000)  executa:  if Usart\_Data\_Ready() > 0 then  Usart\_Read\_Text(recebe,"/")  valorRecebido = StrToInt(recebe)    if (valorRecebido > 999) and (valorRecebido < 4256) then  if (valorRecebido >= 1000) And (valorRecebido < 1256) then  valorRecebido = valorRecebido - 1000  valorPorcent = ((valorRecebido \* 100) / 255)    if valorPorcent > 99 then  valorPorcent = 99  end if  dez = valorPorcent div 10  uni = valorPorcent - (dez \* 10)  valorPortas = dez + (uni \* 16)  escrita = 0x42  gosub escreve\_I2C  end if  if (valorRecebido >= 2000) And (valorRecebido < 2256) then  valorRecebido = valorRecebido - 2000  valorPorcent = ((valorRecebido \* 100) / 255)    if valorPorcent > 99 then  valorPorcent = 99  end if  dez = valorPorcent div 10  uni = valorPorcent - (dez \* 10)  valorPortas = dez + (uni \* 16)  escrita = 0x44  gosub escreve\_I2C  end if  if (valorRecebido >= 3000) And (valorRecebido < 3256) then  valorRecebido = valorRecebido - 3000  valorPorcent = ((valorRecebido \* 100) / 255)    if valorPorcent > 99 then  valorPorcent = 99  end if  dez = valorPorcent div 10  uni = valorPorcent - (dez \* 10)  valorPortas = dez + (uni \* 16)  escrita = 0x46  gosub escreve\_I2C  end if  if (valorRecebido >= 4000) And (valorRecebido < 4256) then  valorRecebido = valorRecebido - 4000  valorPorcent = ((valorRecebido \* 100) / 255)    if valorPorcent > 99 then  valorPorcent = 99  end if  dez = valorPorcent div 10  uni = valorPorcent - (dez \* 10)  valorPortas = dez + (uni \* 16)  escrita = 0x42  gosub escreve\_I2C  escrita = 0x44  gosub escreve\_I2C  escrita = 0x46  gosub escreve\_I2C  end if  end if  Usart\_Write\_Text(recebe)  delay\_ms(5)  USART\_Write\_Text("/")  delay\_ms(5)  end if    goto executa    escreve\_I2C:  I2C\_Start  I2C\_Wr(escrita)  I2C\_Wr(valorPortas)  I2C\_Stop  delay\_ms(10)  return  end. |

**ANEXO - II**

**Esquema elétrico - (Pode ser substituído por esquema impresso).**

|  |
| --- |
|  |