



UNIVERSIDADE FEDERAL DE PERNAMBUCO
CENTRO DE TECNOLOGIA E GEOCIÊNCIAS
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

Circuitos Integrados e Sistemas Embarcados

Relatório Final

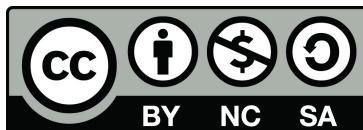
Versão: 1.0.0

Universidade Federal de Pernambuco
Centro de Tecnologia e Geociências/Escola de Engenharia
Programa de Pós-Graduação em Engenharia Elétrica
Departamento de Eletrônica e Sistemas - Bloco B, 4º andar, sala 416.
Avenida da Arquitetura, s/n – Cidade Universitária
50740-550 - Recife - Pernambuco - Brasil
Fone: +55 81 2126-7117 — Fax: +55 81 2126-8215
<https://www.ufpe.br/ppgee>

Recife, 27 de julho de 2015

Licença deste conteúdo

Com o objetivo de promovermos ampla divulgação e possibilitar a participação ativa (de professores, estudantes, egressos, profissionais e da comunidade em geral) no desenvolvimento e aprendizagem da área de Circuitos Integrados e Sistemas Embarcados, disponibilizamos este Relatório Final de Disciplina no Repositório GitHub.



Circuitos Integrados e Sistemas Embarcados - Relatório Final de Gustavo Esteves, João Ferreira, Kádna Maria e Sérgio Mendonça está licenciado com uma Licença Creative Commons – Atribuição (BY) – Não Comercial (NC) – Compartilha Igual (SA) 4.0 Internacional. Baseado no trabalho disponível em <<https://github.com/sftom/ee1054>>.

A equipe.

Equipe de Elaboração e Resolução das Atividades

Elaboração das Atividades

Prof. Marco Aurélio Benedetti Rodrigues
[<http://lattes.cnpq.br/2448324832915432>](http://lattes.cnpq.br/2448324832915432)

Estudantes — Execução das Atividades

Gustavo Ribeiro Porpino Esteves
[<http://lattes.cnpq.br/1140864652601909>](http://lattes.cnpq.br/1140864652601909)

João Ferreira da Silva Júnior
[<http://lattes.cnpq.br/8904695743376784>](http://lattes.cnpq.br/8904695743376784)

Kádna Marina Alves Camboim Vale
[<http://lattes.cnpq.br/2817387497612302>](http://lattes.cnpq.br/2817387497612302)

Sérgio Francisco Tavares de Oliveira Mendonça
[<http://lattes.cnpq.br/6313698968060384>](http://lattes.cnpq.br/6313698968060384)

Lista de ilustrações

Figura 1 – Diagrama Esquemático da Família MCS51 utilizado na Atividade 1	13
Figura 2 – Fluxograma do controle A/D do circuito para utilização total dos canais.	15
Figura 3 – Diagrama de Bloco do Data Sheet do MCS51	16
Figura 4 – Microcontrolador KL Intel P8051.	27
Figura 5 – Executando código em simulador para embarcados Reads51.	28
Figura 6 – Gravador de firmware para Microcontrolador MCS51.	28
Figura 8 – Diagrama de Bloco do Data Sheet do MCS51.	29
Figura 7 – Diagrama Esquemático desenvolvido para o MCS51 80C51 durante a Atividade 2.	30
Figura 9 – Microcontrolador PIC18F4550.	34
Figura 10 – Diagrama Esquemático desenvolvido para o PIC18F4550 durante a Atividade 3.	35
Figura 15 – Diagrama de Bloco do Data Sheet do PIC18F4550.	36
Figura 11 – Programação em IDE MikroC.	37
Figura 12 – Configuração do microcontrolador na IDE MikroC.	37
Figura 13 – Configuração do Bootloader na aplicação PROTO’N.	38
Figura 14 – Gravador de firmware para Microcontrolador PIC18F4550.	38
Figura 20 – Diagrama de Bloco do Data Sheet do MSP430.	46
Figura 16 – Microcontrolador EZ430-RF2500T e seu Gravador de Firmware.	47
Figura 17 – Diagrama Esquemático desenvolvido para o MSP430F2272 durante a Atividade 4.	47
Figura 18 – Programação do microcontrolador MSP430F2272 na IDE CodeComposer.	48
Figura 19 – Programação do microcontrolador MSP430F2272 na IDE CodeComposer.	49
Figura 25 – Diagrama de Bloco do Data Sheet do MSP430.	52
Figura 21 – Diagrama Esquemático desenvolvido para o MSP430F2272 (GRACE) durante a Atividade 5.	53
Figura 22 – Programação da MSP430F2272 na IDE GRACE (ambiente de simulação).	54
Figura 23 – Programação da MSP430F2272 na IDE GRACE (ambiente de simulação).	55
Figura 24 – Programação da MSP430F2272 na IDE GRACE (ambiente de simulação).	56
Figura 28 – Diagrama de Bloco do Data Sheet do Kinetis Freescale.	65
Figura 26 – Plataforma de desenvolvimento Kinetis Freescale utilizada na Atividade 6.	66
Figura 27 – Diagrama Esquemático desenvolvido para o Kinetis Freescale durante a Atividade 6.	66
Figura 29 – Programação da placa Kinetis Freescale FRDM-KL25Z.	67
Figura 33 – Diagrama Pinout da Placa Raspberry Pi B+.	73
Figura 30 – Raspberry B+ utilizada na Atividade 7.	74

Figura 31 – Modelo de LCD utilizado na prática com Raspberry Pi B+.	74
Figura 32 – Diagrama Esquemático desenvolvido para o Raspberry Pi B+ durante a Atividade 7.	74
Figura 34 – Programação da placa Raspberry Pi B+ no ambiente de programação IDLE do Python 3.	75
Figura 35 – Atividade prática na Plataforma de Desenvolvimento Intel Galileo	82
Figura 37 – Diagrama de conexões da Placa Intel Galileo.	83
Figura 36 – Diagrama Esquemático desenvolvido para o Intel Galileo durante a Atividade 8.	84
Figura 38 – Programação da placa Intel Galileo 2nd Gen. através da IDE do Arduino.	85

Lista de tabelas

Tabela 1 – Esquema de programação da da MSP430.	45
---	----

Lista de abreviaturas e siglas

UFPE	Universidade Federal de Pernambuco
PPGEE	Programa de Pós-Graduação em Engenharia Elétrica
CTG	Centro de Tecnologia e Geociências

Sumário

Licença deste conteúdo	2
Equipe de Elaboração e Resolução das Atividades	3
Introdução	10
1 MICROCONTROLADOR MCS51	12
1.1 Atividades propostas	12
1.2 Relatório da Atividade	12
2 MICROCONTROLADOR ATMEL AT89S51	27
2.1 Atividades propostas	27
2.2 Relatório da Atividades	27
3 MICROCONTROLADOR PIC18F4550	34
3.1 Atividades propostas	34
3.2 Relatório da Atividade	34
4 MICROCONTROLADOR MSP430	45
4.1 Atividades propostas	45
4.2 Relatório da Atividade	45
5 MICROCONTROLADOR MSP430 – GRACE	51
5.1 Atividades propostas	51
5.2 Relatório da Atividade	51
6 MICROCONTROLADOR KINETIS FREESCALE FRDM-KL25Z . .	64
6.1 Atividades propostas	64
6.2 Relatório da Atividade	64
7 PLACA RASPBERRY PI B+	72
7.1 Atividades propostas	72
7.2 Relatório da Atividade	72
8 PLACA INTEL GALILEO 2ND GEN.	82
8.1 Atividades propostas	82
8.2 Relatório da Atividade	82

Introdução

A área de Circuitos Integrados e Sistemas Embarcados

De acordo com o atual crescimento tecnológico, pode-se estimar que a era da Internet das Coisas, do Inglês, *Internet of Things* (IoT) demandará por bilhões de novos dispositivos nos próximos 5 anos. Este cenário aponta uma crescente demanda por equipamentos que consumam menos energia, ou que esta seja uma fonte renovável, que executem a transmissão de dados em tempo-real, que disponha de estratégias otimizadas para a manipulação de algoritmos ou de dados, entre outros aspectos. Sem falar da necessidade dos sistemas que interagem uns com os outros.

Requisitos para Sistemas Embarcados

Ao iniciar qualquer projeto, é preciso levantar uma série de questões que possibilitem um melhor entendimento das necessidades, questionar pontos que se mostrem relevantes, e verificar se há concorrência com projetos semelhantes, bem como nos aprofundar nos pontos principais das especificações desejadas.

Uma declaração de escopo precisa ser definida, contendo características principais do sistema, restrições (técnicas, financeiras, comerciais, de tecnologia...), estamos falando especificamente dos requisitos do produto, que envolvem requisitos de hardware e de software (este já pode conter muitas bibliotecas disponíveis, além de entendimentos concisos).

É preciso entender que os requisitos nem sempre são muitos bem especificados, muitas vezes por pouca maturidade no desenvolvimento, por parte do desenvolvedor, muitas vezes por falta da adoção de técnicas ou de uma abordagem sistemática no levantamento desses requisitos.

Independentemente dos caminhos que um projeto venha a tomar, tem-se dois grandes grupos de preocupações:

- O que o sistema deve fazer?
- E, como ele deve implementar todos os requisitos?

A disciplina de Circuitos Integrados e Sistemas Embarcados tem como ementa a análise e projetos com circuitos integrados analógicos e digitais utilizados em instrumentação eletrônica, possibilitando o desenvolvimento de aplicações, utilizando microcontroladores

em sistemas embarcados e circuitos de tempo real. Assim, é possível encontrar as respostas às questões citadas acima, quando se tratando do projeto da disciplina, quanto para projetos de âmbito mercadológico.

Organização deste Relatório

Este relatório final apresenta as atividades realizadas ao longo da Disciplina de Circuitos Integrados e Sistemas Embarcados. Foram desenvolvidas oito atividades, cada uma com propósito específico para determinado tipo de microcontrolador ou placa. Para cada atividade são apresentadas as questões propostas, a resolução de cada uma delas, bem como os códigos em linguagem de programação e algumas considerações. O presente relatório está organizado da seguinte forma: a atividade 1 trata do microcontrolador MCS51, a atividade 2 é sobre o microcontrolador Atmel AT89S51, a atividade 3 apresenta o microcontrolador PIC, a atividade 4 vem com o microcontrolador MSP-430, na atividade 5 temos o microcontrolador MSP430-GRACE, na atividade 6 é apresentado o microcontrolador Kinetis Freescale, já nas atividades 7 e 8 são apresentadas as placas Raspberry Pi B+ e Intel Galileo, respectivamente.

Com o objetivo de colaborar com a comunidade, também disponibilizamos todo o material desenvolvido no GitHub.com, disponível em <<https://github.com/sftom/ee1054>>, para que estes possam servir de orientações e desenvolvimento de estudantes e profissionais da área.

1 Microcontrolador MCS51

1.1 Atividades propostas

Trabalho I – Aula teórica com microcontrolador família MCS51

Atividade 1

Baseado no circuito abaixo realizar o que se pede:

1. Identificar a função do circuito e explicar a função dos componentes eletrônicos utilizados no projeto.
2. Quantos canais estão sendo convertidos para o domínio digital? Proponha o circuito para a utilização dos demais canais do ADC0808.
3. Qual a taxa de amostragem do conversor A/D e quantização utilizada no circuito. Explique.
4. Qual a máxima velocidade permitida para a conversão A/D desse circuito? O que deve ser feito para atingir essa velocidade?
5. E em função do grau de quantização, isto é, da resolução do sinal de tensão convertido, o que se pode fazer para melhorar a qualidade do sinal digital?
6. Faça um programa em C, com a finalidade de:
 - a) Controlar o A/D do circuito conforme o esquemático;
 - b) Controlar o A/D para a utilização total dos canais.

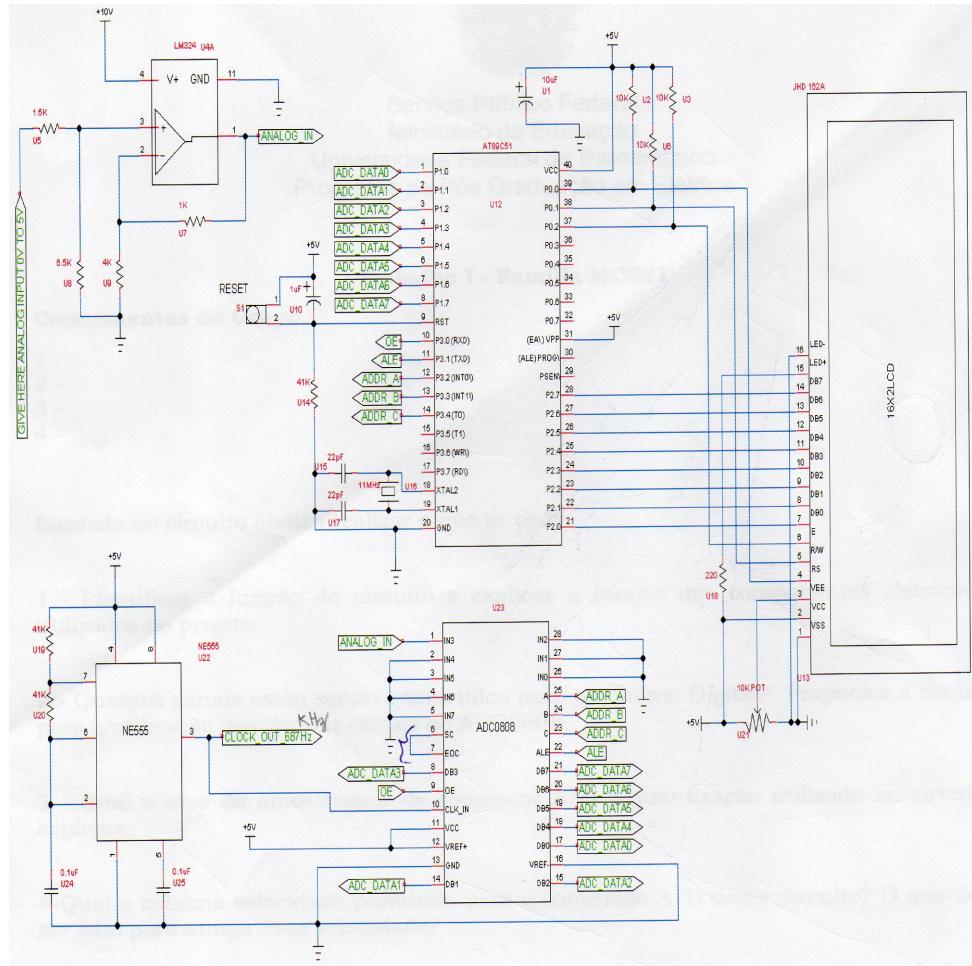
1.2 Relatório da Atividade

As atividades acima são respondidas a seguir:

1. O circuito recebe um sinal analógico a partir do LM324 que amplifica esse sinal e o envia para a entrada IN3 do ADC0808, que faz a amostragem do sinal convertendo de analógico para digital e enviando para o microcontrolador AT89C51, que exibe o sinal amostrado no display 16X21CD, ou seja, recebe o sinal analógico, converte para digital e exibe no LCD.

Figura 1: Diagrama Esquemático da Família MCS51 utilizado na Atividade 1.

Fonte: Produzido pelo Prof. Marco Aurélio Benedetti Rodrigues



2. Apenas um sinal está sendo convertido para o domínio digital IN3 do ADC0808. A proposta para o circuito é replicar o circuito de aquisição do LM324 para cada uma das portas disponíveis do conversor A/D (IN4, IN5, IN6, IN7, IN2, IN1 e IN0) e em seguida considerar a informação dos pinos de endereçamento (ADDR_A, ADDR_B e ADDR_C) que informa qual a porta de entrada do sinal analógico junto com a informação da porta ALE, que sinaliza a finalização do endereço novo.
3. A taxa de amostragem do conversor A/D é 10khz, pois seu tempo de conversão é de 100 microsssegundos. Mas, como o circuito A/D utiliza apenas uma porta e possui o clock de 687khz, a taxa de amostragem será de 10,735khz. A quantização utilizada no circuito é de 8 bits, o que dá 255 níveis.
4. A velocidade máxima permitida se obtém elevando-se o clock para 1280khz e curto-circuitar os pinos SC e EOC do conversor A/D, o que já está feito no circuito.
5. Para melhorar a qualidade do sinal digital deve-se ajustar os limites de saída do LM324 de acordo com os limites de entrada do ADC0808.

6. A descrição do fluxo do programa pode ser observado a seguir, bem como um pseudocódigo, desenvolvido em Linguagem C.

- a) O MC envia o endereço a ser lido pelas portas ADDR_A, ADDR_B e ADDR_C, com o valor de A=0, B=1 e C=1 que corresponde a porta ANALOG_IN3 do conversor A/D. Após isso, aciona o latch em nível alto, dá um delay de 200 ms para aguardar a conversão pelo A/D e aciona o pino OE para habilitar a leitura do valor de saída, executa a leitura. Os pinos SC e EOC não são controlados, pois, estão curto-circuitados.
- b) O MC envia o endereço a ser lido pelas portas ADDR_A, ADDR_B e ADDR_C com seus valores variando de 000 até 111, correspondendo às portas ANALOG_IN[0-7] do conversor A/D, para cada endereço repete-se o seguinte processo: aciona o latch em nível alto, dá um delay de 200ms e enviar o latch (ALE), para o nível baixo, novamente da outro delay de 200ms para aguardar a conversão A/D e aciona o pino OE para habilitar a leitura do valor de saída, executa a leitura. A partir daqui segue para o próximo endereço ANALOG_IN, do início desse processo. Os pinos SC e EOC não são controlados, pois estão curto-circuitados.

Para fonte de consulta, inserimos o Diagrama de Bloco, como pode ser observado na Figura 3, extraída do Data Sheet do MCS51. ([CORP., 2000](#)).

Podemos observar um fluxo do programa na Figura 2, com a ideia de melhorar o entendimento do programa, e a partir deste, construir o código em Linguagem C.

Figura 2: Fluxograma do controle A/D do circuito para utilização total dos canais.

Fonte: Produzido pelos autores.

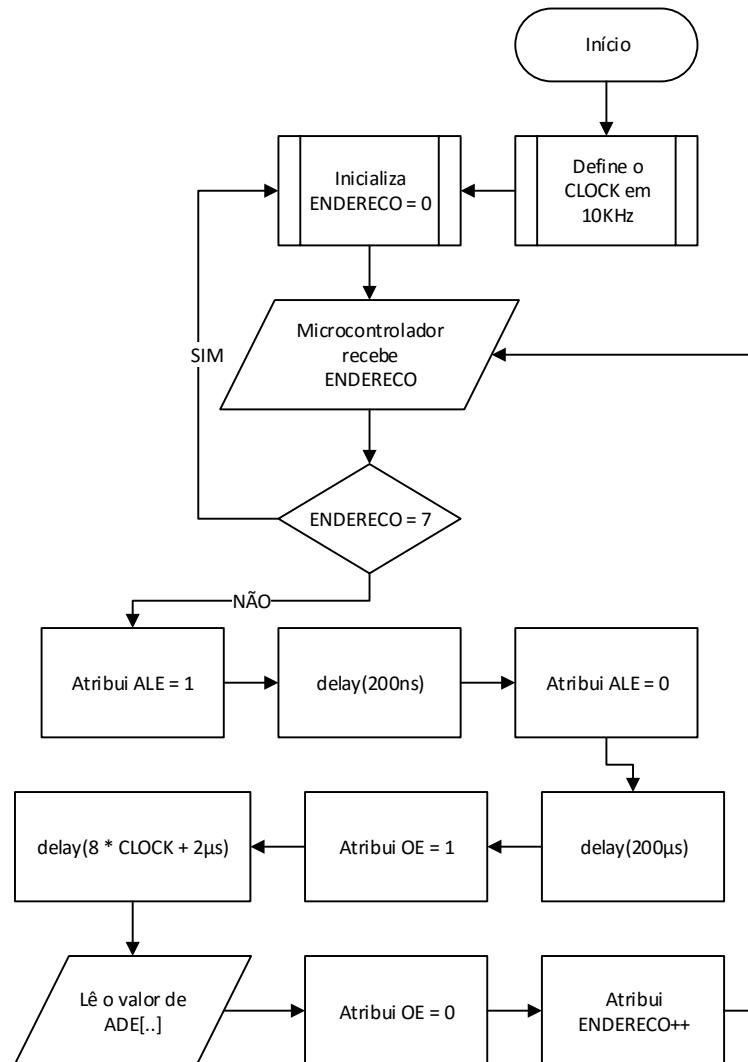
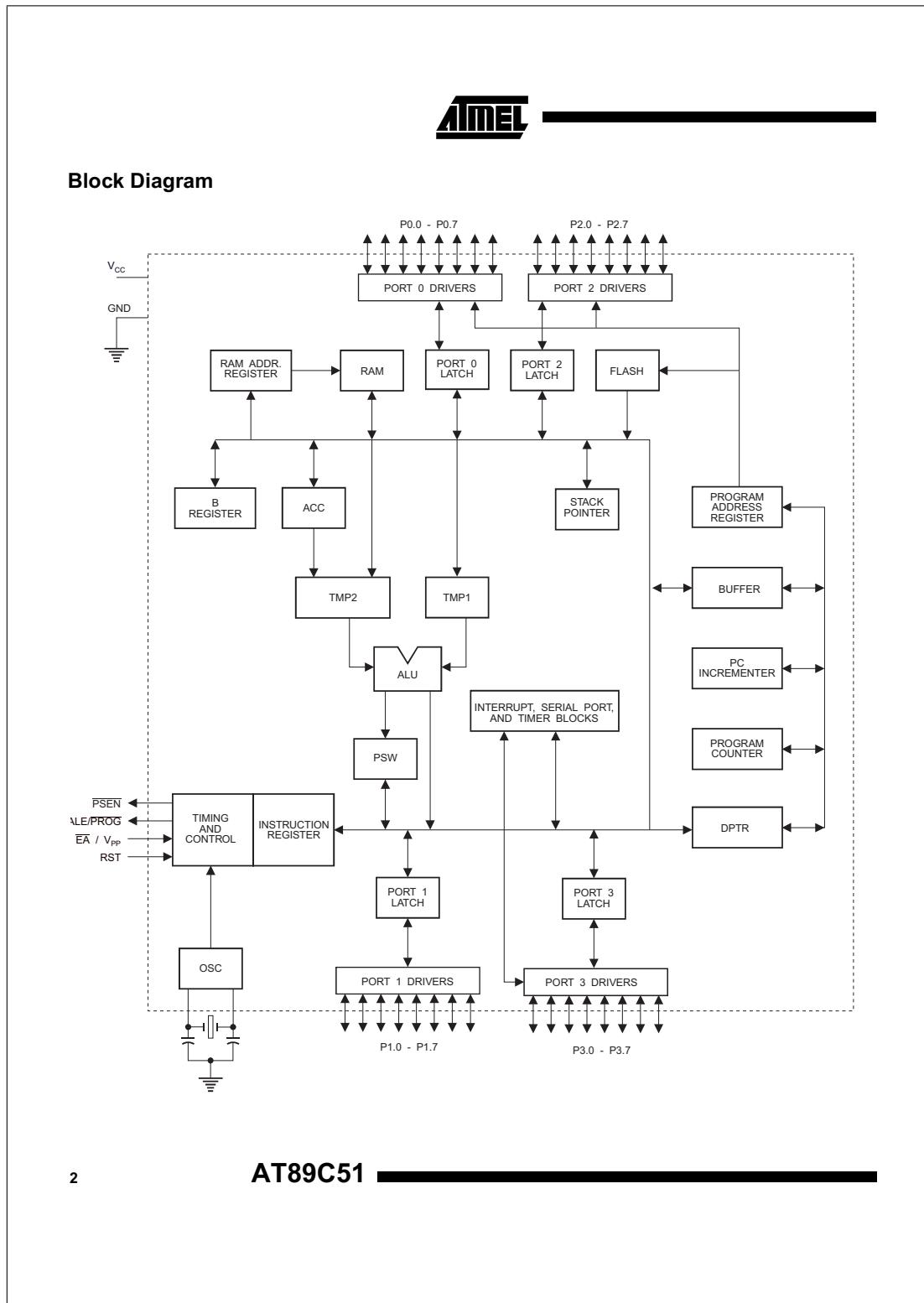


Figura 3: Diagrama de Bloco do Data Sheet do MCS51



Os pseudocódigos elaborados estão disponibilizados a seguir:

Microcontrolador MCS51 80C51 – A01 (Pseudocódigo)

A01Q06a

```
1 algoritmo "EE1054-Atividade01-6A"
2 // Licenca Creative Commons
3 // Circuitos Integrados e Sistemas Embarcados - Relatorio Final de
4 // Gustavo Esteves, Joao Ferreira, Kadna Maria e Sergio Mendonca
5 // esta licenciado com uma Licenca Creative Commons
6 // Atribuicao-NaoComercial-CompartilhaIgual 4.0 Internacional.
7 // Baseado no trabalho disponivel em https://github.com/sftom/ee1054
8 //
9 // +-----+
10 // | Universidade Federal de Pernambuco | |
11 // | Programa de Pos-graduacao em Engenharia Eletrica | |
12 // +-----+
13 // +-----+
14 // | EE1054 Circuitos Integrados e Sistemas Embarcados | |
15 // | Turma 2015.1 | |
16 // +-----+
17 // Alunos em ordem alfabetica:
18 // > Gustavo Ribeiro Porpino Esteves
19 // > Joao Ferreira da Silva Junior
20 // > Kadna Maria Alves Camboim Vale
21 // > Sergio Francisco Tavares de Oliveira Mendonca
22 // +-----+
23 var
24     clock: inteiro
25     // Portas do Microcontrolador
26     // Portas de enderecamento
27     P32, P33, P34: caractere
28     // Portas de entrada digital
29     P10, P11, P12, P13, P14, P15, P16, P17 : inteiro
30     // Portas OE(RXD) e ALE(TXD)
31     P30, P31: inteiro
32     // Portas do conversor A/D
33     // Portas de entrada analogica
34     IN0, IN1, IN2, IN3, IN4, IN5, IN6, IN7: real
35     // Portas de enderecamento
36     A, B, C: caractere
37     // Portas de saida digital
38     DB0, DB1, DB2, DB3, DB4, DB5, DB6, DB7: inteiro
39     // Vetor comum de enderecamento
40     ADDR: vetor [1..3] de caractere
41     // Variaveis comuns de iteracao
```

```
42     i, j, k: inteiro
43 inicio
44     // Atribui valores aleatorios para as entradas analogicas
45     IN0 <- (exp(10 * rand, rand)) * 10
46     IN1 <- (exp(10 * rand, rand)) * 10
47     IN2 <- (exp(10 * rand, rand)) * 10
48     IN3 <- (exp(10 * rand, rand)) * 10
49     IN4 <- (exp(10 * rand, rand)) * 10
50     IN5 <- (exp(10 * rand, rand)) * 10
51     IN6 <- (exp(10 * rand, rand)) * 10
52     IN7 <- (exp(10 * rand, rand)) * 10
53     // Define o clock em 10KHz
54     clock <- 10
55     // O Microcontrolador envia o endereco a ser lido para o conversor A/D
56     i <- 0
57     j <- 1
58     k <- 1
59     P32 <- numpcarac(i)
60     P33 <- numpcarac(j)
61     P34 <- numpcarac(k)
62     A <- P32
63     B <- P33
64     C <- P34
65     ADDR[1] <- A
66     ADDR[2] <- B
67     ADDR[3] <- C
68     escolha (ADDR[1] + ADDR[2] + ADDR[3])
69     caso ("011")
70         P31 <- 1
71         timer 2
72         P31 <- 0
73         timer 2
74         P30 <- 1
75         timer (8 * clock)
76         DB3 <- int(IN3)
77         P30 <- 0
78         timer 1
79 fimescolha
80
81 timer 10
82 limpatela
83 escreval("-----+")
84 escreval(" | Universidade Federal de Pernambuco | ")
85 escreval(" | Programa de Pos-graduacao em Engenharia Eletrica | ")
86 escreval("-----+")
87 escreval("-----+")
88 escreval(" | EE1054 Circuitos Integrados e Sistemas Embarcados | ")
```

```

89 escreval(" | Turma 2015.1 | ")
90 escreval(" +-----+")
91 escreval(" Alunos em ordem alfabetica:")
92 escreval(" > Gustavo Ribeiro Porpino Esteves")
93 escreval(" > Joao Ferreira da Silva Junior")
94 escreval(" > Kadna Maria Alves Camboim Vale")
95 escreval(" > Sergio Francisco Tavares de Oliveira Mendonca")
96 escreval(" +-----+")
97 escreval(""))
98 escreval("Display de LCD 16x2")
99 escreval(" +-----+")
100 escreval(" VALOR LIDO DB3 ")
101 escreval(" >> " + numpcarac(DB3))
102 escreval(" +-----+")
103 timer 1000
104
105 fimalgoritmo

```

sources/EE1054-Atividade01-6A.alg

A01Q06b

```

1 algoritmo "EE1054-Atividade01-6B"
2 // Licenca Creative Commons
3 // Circuitos Integrados e Sistemas Embarcados - Relatorio Final de
4 // Gustavo Esteves, Joao Ferreira, Kadna Maria e Sergio Mendonca
5 // esta licenciado com uma Licenca Creative Commons
6 // Atribuicao-NaoComercial-CompartilhaIgual 4.0 Internacional.
7 // Baseado no trabalho disponivel em https://github.com/sftom/ee1054
8 //
9 // +-----+
10 // | Universidade Federal de Pernambuco | |
11 // | Programa de Pos-graduacao em Engenharia Eletrica | |
12 // +-----+
13 // +-----+
14 // | EE1054 Circuitos Integrados e Sistemas Embarcados | |
15 // | Turma 2015.1 | |
16 // +-----+
17 // | Alunos em ordem alfabetica: | |
18 // | > Gustavo Ribeiro Porpino Esteves | |
19 // | > Joao Ferreira da Silva Junior | |
20 // | > Kadna Maria Alves Camboim Vale | |
21 // | > Sergio Francisco Tavares de Oliveira Mendonca | |
22 // +-----+
23 var
24     clock: inteiro
25     // Portas do Microcontrolador

```

```
26 // Portas de enderecamento
27 P32, P33, P34: caractere
28 // Portas de entrada digital
29 P10, P11, P12, P13, P14, P15, P16, P17 : inteiro
30 // Portas OE(RXD) e ALE(TXD)
31 P30, P31: inteiro
32 // Portas do conversor A/D
33 // Portas de entrada analogica
34 IN0, IN1, IN2, IN3, IN4, IN5, IN6, IN7: real
35 // Portas de enderecamento
36 A, B, C: caractere
37 // Portas de saida digital
38 DB0, DB1, DB2, DB3, DB4, DB5, DB6, DB7: inteiro
39 // Vetor comum de enderecamento
40 ADDR: vetor [1..3] de caractere
41 // Variaveis comuns de iteracao
42 i, j, k: inteiro
43 inicio
44 // Atribui valores aleatorios para as entradas analogicas
45 IN0 <- (exp(10 * rand, rand)) * 100
46 IN1 <- (exp(10 * rand, rand)) * 100
47 IN2 <- (exp(10 * rand, rand)) * 100
48 IN3 <- (exp(10 * rand, rand)) * 100
49 IN4 <- (exp(10 * rand, rand)) * 100
50 IN5 <- (exp(10 * rand, rand)) * 100
51 IN6 <- (exp(10 * rand, rand)) * 100
52 IN7 <- (exp(10 * rand, rand)) * 100
53 // Define o clock em 10KHz
54 clock <- 10
55 // O Microcontrolador envia o endereco a ser lido para o conversor A/D
56 para i de 0 ate 1 faca
57     para j de 0 ate 1 faca
58         para k de 0 ate 1 faca
59     P32 <- numpcarac(i)
60     P33 <- numpcarac(j)
61     P34 <- numpcarac(k)
62     A <- P32
63     B <- P33
64     C <- P34
65     ADDR[1] <- A
66     ADDR[2] <- B
67     ADDR[3] <- C
68 escolha (ADDR[1] + ADDR[2] + ADDR[3])
69     caso ("000")
70         P31 <- 1
71         timer 2
72         P31 <- 0
```

```
73     timer 2
74     P30 <- 1
75     timer (8 * clock)
76     DB0 <- int(IN0)
77     P30 <- 0
78     timer 1
79     caso ("001")
80         P31 <- 1
81         timer 2
82         P31 <- 0
83         timer 2
84         P30 <- 1
85         timer (8 * clock)
86         DB1 <- int(IN1)
87         P30 <- 0
88         timer 1
89     caso ("010")
90         P31 <- 1
91         timer 2
92         P31 <- 0
93         timer 2
94         P30 <- 1
95         timer (8 * clock)
96         DB2 <- int(IN2)
97         P30 <- 0
98         timer 1
99     caso ("011")
100        P31 <- 1
101        timer 2
102        P31 <- 0
103        timer 2
104        P30 <- 1
105        timer (8 * clock)
106        DB3 <- int(IN3)
107        P30 <- 0
108        timer 1
109    caso ("100")
110        P31 <- 1
111        timer 2
112        P31 <- 0
113        timer 2
114        P30 <- 1
115        timer (8 * clock)
116        DB4 <- int(IN4)
117        P30 <- 0
118        timer 1
119    caso ("101")
```

```
120      P31 <- 1
121      timer 2
122      P31 <- 0
123      timer 2
124      P30 <- 1
125      timer (8 * clock)
126      DB5 <- int(IN5)
127      P30 <- 0
128      timer 1
129  caso ("110")
130      P31 <- 1
131      timer 2
132      P31 <- 0
133      timer 2
134      P30 <- 1
135      timer (8 * clock)
136      DB6 <- int(IN6)
137      P30 <- 0
138      timer 1
139  caso ("111")
140      P31 <- 1
141      timer 2
142      P31 <- 0
143      timer 2
144      P30 <- 1
145      timer (8 * clock)
146      DB7 <- int(IN7)
147      P30 <- 0
148      timer 1
149 fimescolha
150      fimpala
151      fimpala
152 fimpala
153
154 timer 10
155 limpatela
156 escreval("-----+")
157 escreval(" | Universidade Federal de Pernambuco | ")
158 escreval(" | Programa de Pos-graduacao em Engenharia Eletrica | ")
159 escreval("-----+")
160 escreval("-----+")
161 escreval(" | EE1054 Circuitos Integrados e Sistemas Embarcados | ")
162 escreval(" | Turma 2015.1 | ")
163 escreval("-----+")
164 escreval(" Alunos em ordem alfabetica:")
165 escreval(" > Gustavo Ribeiro Porpino Esteves")
166 escreval(" > Joao Ferreira da Silva Junior")
```

```
167 escreval(" > Kadna Maria Alves Camboim Vale")
168 escreval(" > Sergio Francisco Tavares de Oliveira Mendonca")
169 escreval("-----+")
170 escreval("")
171 escreval("Display de LCD 16x2")
172 escreval("-----+")
173 escreval(" VALOR LIDO DB0 ")
174 escreval(" >> " + numpcarac(DB0))
175 escreval("-----+")
176 timer 1000
177
178 timer 10
179 limpatela
180 escreval("-----+")
181 escreval(" | Universidade Federal de Pernambuco | ")
182 escreval(" | Programa de Pos-graduacao em Engenharia Eletrica | ")
183 escreval("-----+")
184 escreval("-----+")
185 escreval(" | EE1054 Circuitos Integrados e Sistemas Embarcados | ")
186 escreval(" | Turma 2015.1 | ")
187 escreval("-----+")
188 escreval(" Alunos em ordem alfabetica:")
189 escreval(" > Gustavo Ribeiro Porpino Esteves")
190 escreval(" > Joao Ferreira da Silva Junior")
191 escreval(" > Kadna Maria Alves Camboim Vale")
192 escreval(" > Sergio Francisco Tavares de Oliveira Mendonca")
193 escreval("-----+")
194 escreval("")
195 escreval("Display de LCD 16x2")
196 escreval("-----+")
197 escreval(" VALOR LIDO DB1 ")
198 escreval(" >> " + numpcarac(DB1))
199 escreval("-----+")
200 timer 1000
201
202 timer 10
203 limpatela
204 escreval("-----+")
205 escreval(" | Universidade Federal de Pernambuco | ")
206 escreval(" | Programa de Pos-graduacao em Engenharia Eletrica | ")
207 escreval("-----+")
208 escreval("-----+")
209 escreval(" | EE1054 Circuitos Integrados e Sistemas Embarcados | ")
210 escreval(" | Turma 2015.1 | ")
211 escreval("-----+")
212 escreval(" Alunos em ordem alfabetica:")
213 escreval(" > Gustavo Ribeiro Porpino Esteves")
```

```
214 escreval(" > Joao Ferreira da Silva Junior")
215 escreval(" > Kadna Maria Alves Camboim Vale")
216 escreval(" > Sergio Francisco Tavares de Oliveira Mendonca")
217 escreval("-----+")
218 escreval(""))
219 escreval("Display de LCD 16x2")
220 escreval("-----+")
221 escreval(" VALOR LIDO DB2 ")
222 escreval(" >> " + numpcarac(DB2))
223 escreval("-----+")
224 timer 1000
225
226 timer 10
227 limpatela
228 escreval("-----+")
229 escreval(" | Universidade Federal de Pernambuco | ")
230 escreval(" | Programa de Pos-graduacao em Engenharia Eletrica | ")
231 escreval("-----+")
232 escreval("-----+")
233 escreval(" | EE1054 Circuitos Integrados e Sistemas Embarcados | ")
234 escreval(" | Turma 2015.1 | ")
235 escreval("-----+")
236 escreval(" Alunos em ordem alfabetica:")
237 escreval(" > Gustavo Ribeiro Porpino Esteves")
238 escreval(" > Joao Ferreira da Silva Junior")
239 escreval(" > Kadna Maria Alves Camboim Vale")
240 escreval(" > Sergio Francisco Tavares de Oliveira Mendonca")
241 escreval("-----+")
242 escreval(""))
243 escreval("Display de LCD 16x2")
244 escreval("-----+")
245 escreval(" VALOR LIDO DB3 ")
246 escreval(" >> " + numpcarac(DB3))
247 escreval("-----+")
248 timer 1000
249
250 timer 10
251 limpatela
252 escreval("-----+")
253 escreval(" | Universidade Federal de Pernambuco | ")
254 escreval(" | Programa de Pos-graduacao em Engenharia Eletrica | ")
255 escreval("-----+")
256 escreval("-----+")
257 escreval(" | EE1054 Circuitos Integrados e Sistemas Embarcados | ")
258 escreval(" | Turma 2015.1 | ")
259 escreval("-----+")
260 escreval(" Alunos em ordem alfabetica:")
```

```
261 escreval(" > Gustavo Ribeiro Porpino Esteves")
262 escreval(" > Joao Ferreira da Silva Junior")
263 escreval(" > Kadna Maria Alves Camboim Vale")
264 escreval(" > Sergio Francisco Tavares de Oliveira Mendonca")
265 escreval("-----+")
266 escreval(""))
267 escreval("Display de LCD 16x2")
268 escreval("-----+")
269 escreval(" VALOR LIDO DB4 ")
270 escreval(" >> " + numpcarac(DB4) )
271 escreval("-----+")
272 timer 1000
273
274 timer 10
275 limpatela
276 escreval("-----+")
277 escreval(" | Universidade Federal de Pernambuco | ")
278 escreval(" | Programa de Pos-graduacao em Engenharia Eletrica | ")
279 escreval("-----+")
280 escreval("-----+")
281 escreval(" | EE1054 Circuitos Integrados e Sistemas Embarcados | ")
282 escreval(" | Turma 2015.1 | ")
283 escreval("-----+")
284 escreval(" Alunos em ordem alfabetica:")
285 escreval(" > Gustavo Ribeiro Porpino Esteves")
286 escreval(" > Joao Ferreira da Silva Junior")
287 escreval(" > Kadna Maria Alves Camboim Vale")
288 escreval(" > Sergio Francisco Tavares de Oliveira Mendonca")
289 escreval("-----+")
290 escreval("")
291 escreval("Display de LCD 16x2")
292 escreval("-----+")
293 escreval(" VALOR LIDO DB5 ")
294 escreval(" >> " + numpcarac(DB5) )
295 escreval("-----+")
296 timer 1000
297
298 timer 10
299 limpatela
300 escreval("-----+")
301 escreval(" | Universidade Federal de Pernambuco | ")
302 escreval(" | Programa de Pos-graduacao em Engenharia Eletrica | ")
303 escreval("-----+")
304 escreval("-----+")
305 escreval(" | EE1054 Circuitos Integrados e Sistemas Embarcados | ")
306 escreval(" | Turma 2015.1 | ")
307 escreval("-----+")
```

```
308 escreval(" Alunos em ordem alfabetica:")
309 escreval(" > Gustavo Ribeiro Porpino Esteves")
310 escreval(" > Joao Ferreira da Silva Junior")
311 escreval(" > Kadna Maria Alves Camboim Vale")
312 escreval(" > Sergio Francisco Tavares de Oliveira Mendonca")
313 escreval("-----+")
314 escreval(""))
315 escreval("Display de LCD 16x2")
316 escreval("-----+")
317 escreval(" VALOR LIDO DB6 ")
318 escreval(" >> " + numpcarac(DB6))
319 escreval("-----+")
320 timer 1000
321
322 timer 10
323 limpatela
324 escreval("-----+")
325 escreval(" | Universidade Federal de Pernambuco | ")
326 escreval(" | Programa de Pos-graduacao em Engenharia Eletrica | ")
327 escreval("-----+")
328 escreval("-----+")
329 escreval(" | EE1054 Circuitos Integrados e Sistemas Embarcados | ")
330 escreval(" | Turma 2015.1 | ")
331 escreval("-----+")
332 escreval(" Alunos em ordem alfabetica:")
333 escreval(" > Gustavo Ribeiro Porpino Esteves")
334 escreval(" > Joao Ferreira da Silva Junior")
335 escreval(" > Kadna Maria Alves Camboim Vale")
336 escreval(" > Sergio Francisco Tavares de Oliveira Mendonca")
337 escreval("-----+")
338 escreval(""))
339 escreval("Display de LCD 16x2")
340 escreval("-----+")
341 escreval(" VALOR LIDO DB7 ")
342 escreval(" >> " + numpcarac(DB7))
343 escreval("-----+")
344 timer 1000
345
346 fimalgoritmo
```

2 Microcontrolador ATMEIL AT89S51

2.1 Atividades propostas

Relatório de experimentos – Aula prática com microcontrolador família MCS51

Atividade 2

1. Desenvolver um programa para realizar as seguintes funções:
 - a) Ao pressionar uma tecla acender e apagar todos os quatro LEDs da placa.
 - b) Ao pressionar a segunda tecla fazer com que os quatro LEDs se acendam em ordem crescente.
 - c) Ao pressionar a terceira tecla fazer com que os quatro LEDs se acendam em ordem decrescente.
 - d) Ao pressionar a quarta tecla realizar o item a, b e c em sequência.

2.2 Relatório da Atividades

A seguir, podemos visualizar uma imagem do microcontrolador MCS51 utilizado para a realização dos experimentos.

Figura 4: Microcontrolador KL Intel P8051.



Para essa atividade foi utilizado o Reads51 que permite simular as entradas e saídas do controlador através de botões mostrados na Figura 5.

A gravação dos códigos desenvolvidos no MCS51 é realizada através de um gravador, como mostrado na Figura 6.

Conforme solicitado na Atividade 2 foi necessária a utilização de uma placa de circuito cedido para as equipes, que continha os componentes que seria utilizados, como

Figura 5: Executando código em simulador para embarcados Reads51.

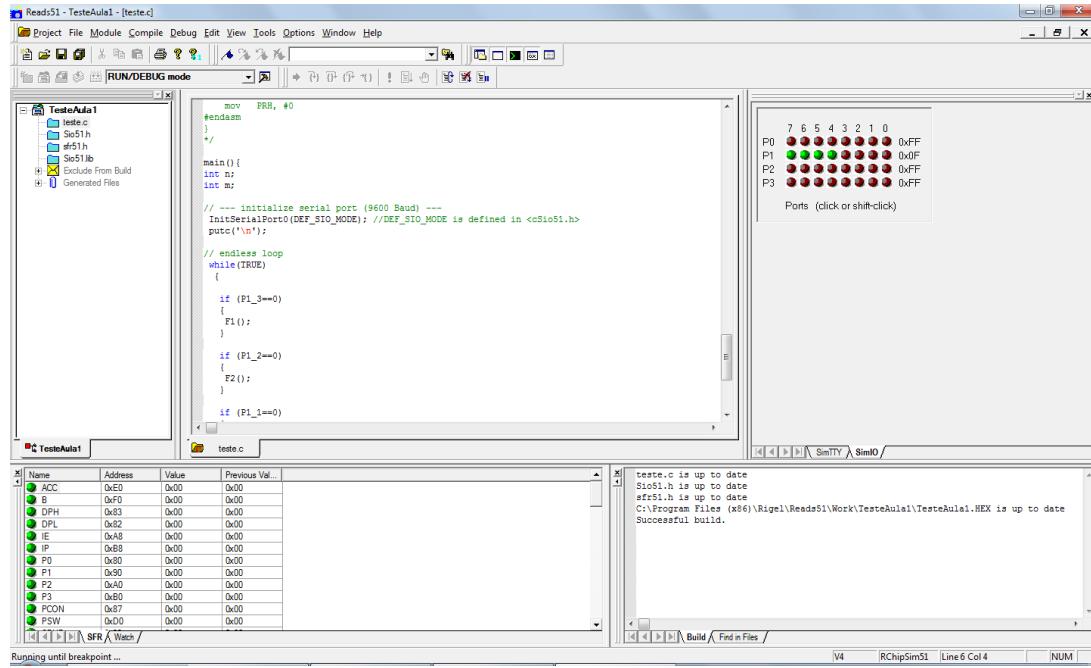
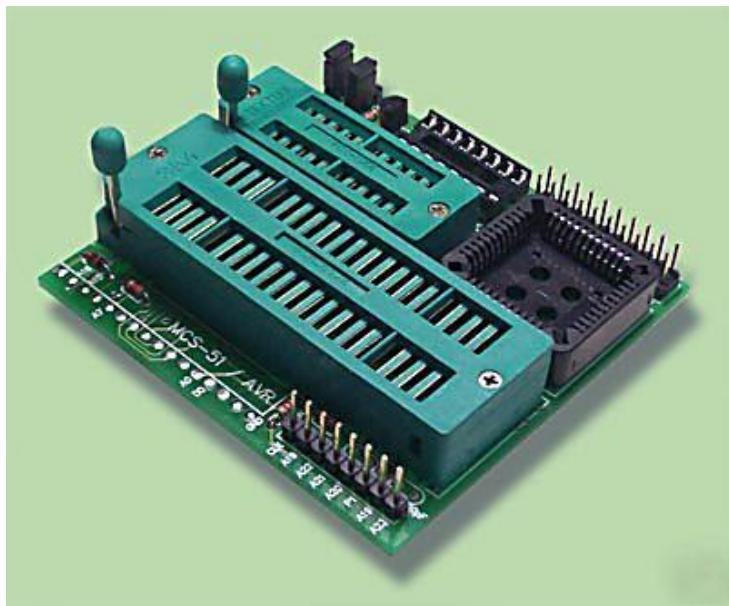


Figura 6: Gravador de firmware para Microcontrolador MCS51.



por exemplo, botões e LED's. O Diagrama Esquemático das ligações realizadas na placa pode ser observado na Figura 7.

Como trabalhamos, teoricamente, no mesmo microcontrolador da Atividade 1, inserimos novamente o Diagrama de Bloco, como pode ser observado na Figura 3, extraída do Data Sheet do Atmel AT89S51. (CORP., 2000).

Figura 8: Diagrama de Bloco do Data Sheet do MCS51.

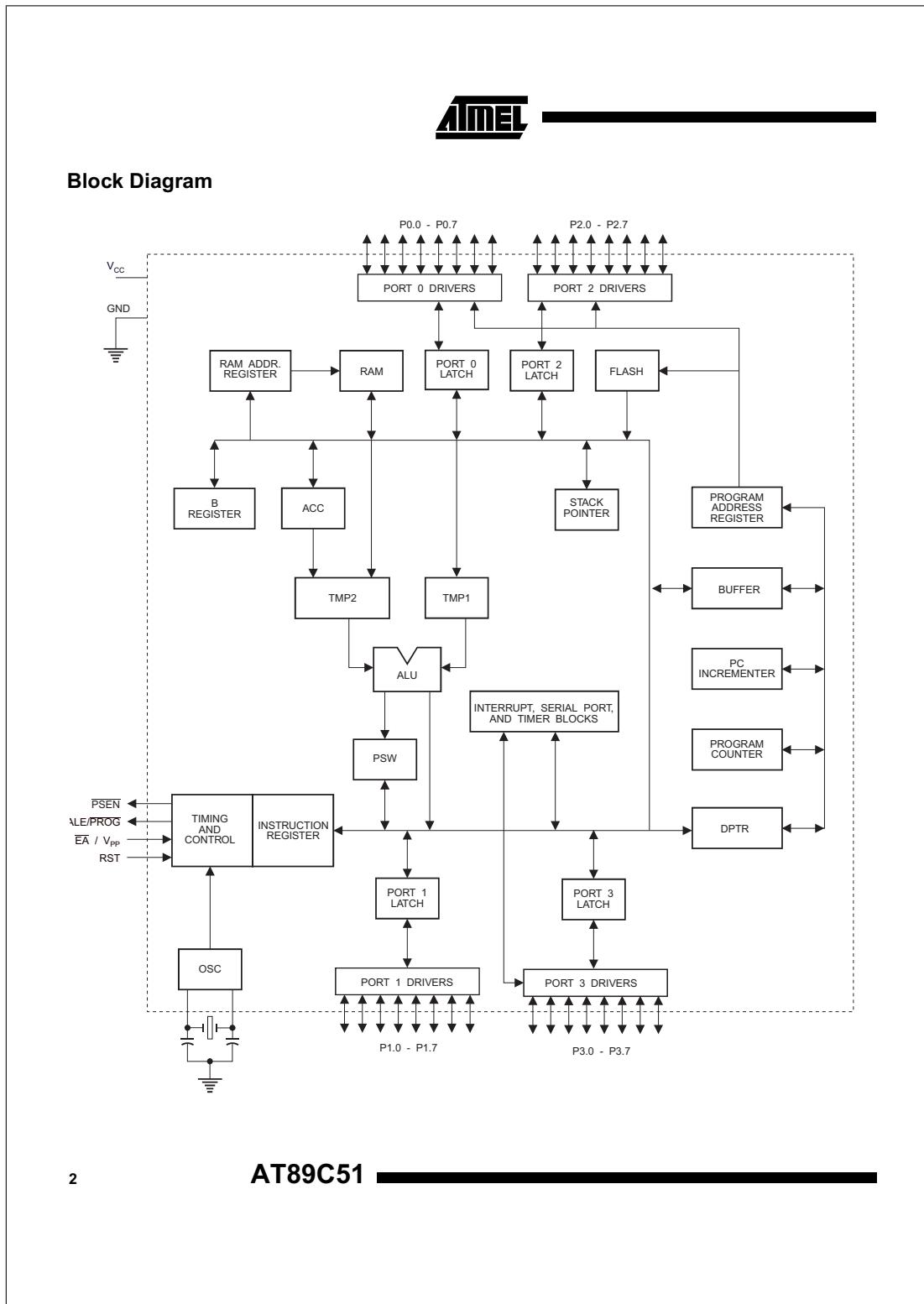
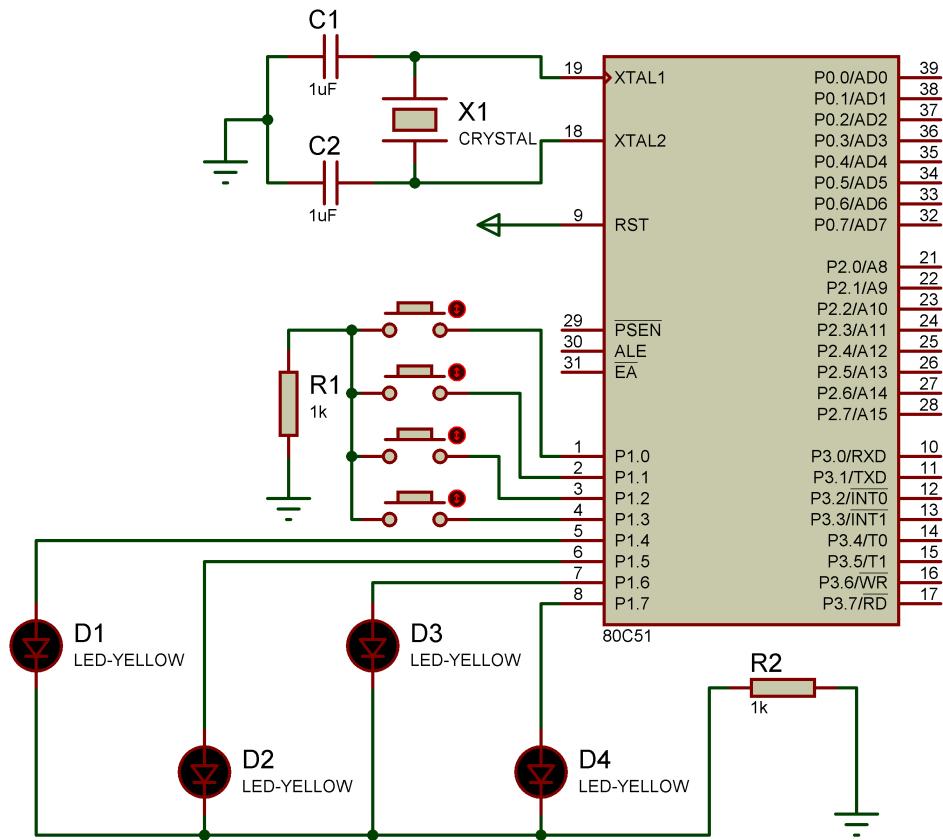


Figura 7: Diagrama Esquemático desenvolvido para o MCS51 80C51 durante a Atividade 2.

Fonte: Produzido pelos autores.



Os códigos-fontes elaborados estão disponibilizados a seguir:

Microcontrolador ATMEL AT89S51 – A02 (Códigos-fontes)

A02Q01

```

1 // Licenca Creative Commons
2 // Circuitos Integrados e Sistemas Embarcados - Relatorio Final de
3 // Gustavo Esteves, Joao Ferreira, Kadna Maria e Sergio Mendonca
4 // esta licenciado com uma Licenca Creative Commons
5 // Atribuicao-NaoComercial-CompartilhaIgual 4.0 Internacional.
6 // Baseado no trabalho disponivel em https://github.com/sftom/ee1054
7 //
8 // +-----+
9 // | Universidade Federal de Pernambuco | |
10 // | Programa de Pos-graduacao em Engenharia Eletrica | |
11 // +-----+
12 // +-----+
13 // | EE1054 Circuitos Integrados e Sistemas Embarcados | |
14 // | Turma 2015.1 | |

```

```
15 // +-----+
16 // Alunos em ordem alfabetica:
17 // > Gustavo Ribeiro Porpino Esteves
18 // > Joao Ferreira da Silva Junior
19 // > Kadna Maria Alves Camboim Vale
20 // > Sergio Francisco Tavares de Oliveira Mendonca
21 // +-----+
22 #include <Sio51.h>
23 #include <sfr51.h>
24 /*
25 // Desenvolver um programa para realizar as seguintes funcoes:
26 // a) Ao pressionar uma tecla acender e depois apagar todos os quatro
27 // LEDs da placa.
28 // b) Ao pressionar a segunda tecla fazer com que os 4 LEDs se acendam
29 // em ordem crescente.
30 // c) Ao pressionar a terceira tecla fazer com que os LEDs se acendam
31 // em ordem decrescente.
32 // d) Ao pressionar a quarta tecla realizar os itens A, B e C em
33 // sequencia.
34 */
35 #define TRUE 1
36 #define FALSE 0
37
38 void delay(void) {
39     int n;
40     int m;
41     for(n=0; n<200; n++) {
42         for(m=0; m<1000; m++);
43     }
44 }
45
46 void F1(void) {
47     #asm
48     clr P1.7
49     clr P1.6
50     clr P1.5
51     clr P1.4
52     #endasm
53     putc('+');
54     delay();
55     #asm
56     setb P1.7
57     setb P1.6
58     setb P1.5
59     setb P1.4
60     #endasm
61     delay();
```

```
62 }
63
64 void F2(void) {
65     #asm
66     clr P1.7
67     #endasm
68     delay();
69     #asm
70     setb P1.7
71     #endasm
72     #asm
73     clr P1.6
74     #endasm
75     delay();
76     #asm
77     setb P1.6
78     #endasm
79     #asm
80     clr P1.5
81     #endasm
82     delay();
83     #asm
84     setb P1.5
85     #endasm
86     #asm
87     clr P1.4
88     #endasm
89     delay();
90     #asm
91     setb P1.4
92     #endasm
93 }
94
95 void F3(void) {
96     #asm
97     clr P1.4
98     #endasm
99     delay();
100    #asm
101    setb P1.4
102    #endasm
103    #asm
104    clr P1.5
105    #endasm
106    delay();
107    #asm
108    setb P1.5
```

```
109 #endasm
110 #asm
111 clr P1.6
112 #endasm
113 delay();
114 #asm
115 setb P1.6
116 #endasm
117 #asm
118 clr P1.7
119 #endasm
120 delay();
121 #asm
122 setb P1.7
123 #endasm
124 }
125
126 int main() {
127     int n;
128     int m;
129     InitSerialPort0(DEF_SIO_MODE);
130     putc('\n');
131     while(TRUE) {
132         if (P1_3==0) {
133             F1();
134         }
135         if (P1_2==0) {
136             F2();
137         }
138         if (P1_1==0) {
139             F3();
140         }
141         if (P1_0==0) {
142             F1();
143             F2();
144             F3();
145         }
146     }
147 }
```

sources/EE1054-Atividade02.c

3 Microcontrolador PIC18F4550

3.1 Atividades propostas

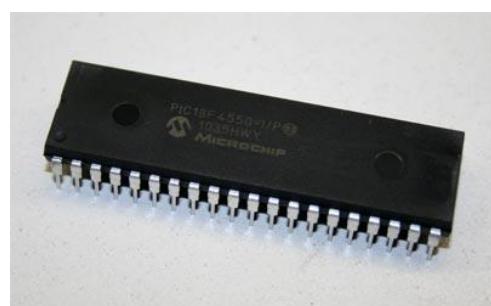
Relatório de experimentos – Aula prática com microcontrolador família PIC

Atividade 3

1. Instalar o programa para bootloader e o simulador cruzado MicroC: InstallPROTON e microc-pro-pic640.
2. Fazer um programa para acionar dois LEDs da placa PIC fornecida. Utilizar o arquivo Manual PROTON para identificar os LEDs e onde estão conectados.
3. Ao pressionar uma tecla, acender e depois apagar todos os dois LEDs da placa.
4. Ao pressionar pela segunda vez a tecla, fazer com que os 2 LEDs se acendam da seguinte forma:
 - a) LED 1 acende e apaga a cada 100ms
 - b) Após 10 piscadas do LED1 o LED2 fica aceso por 100ms.

3.2 Relatório da Atividade

Figura 9: Microcontrolador PIC18F4550.



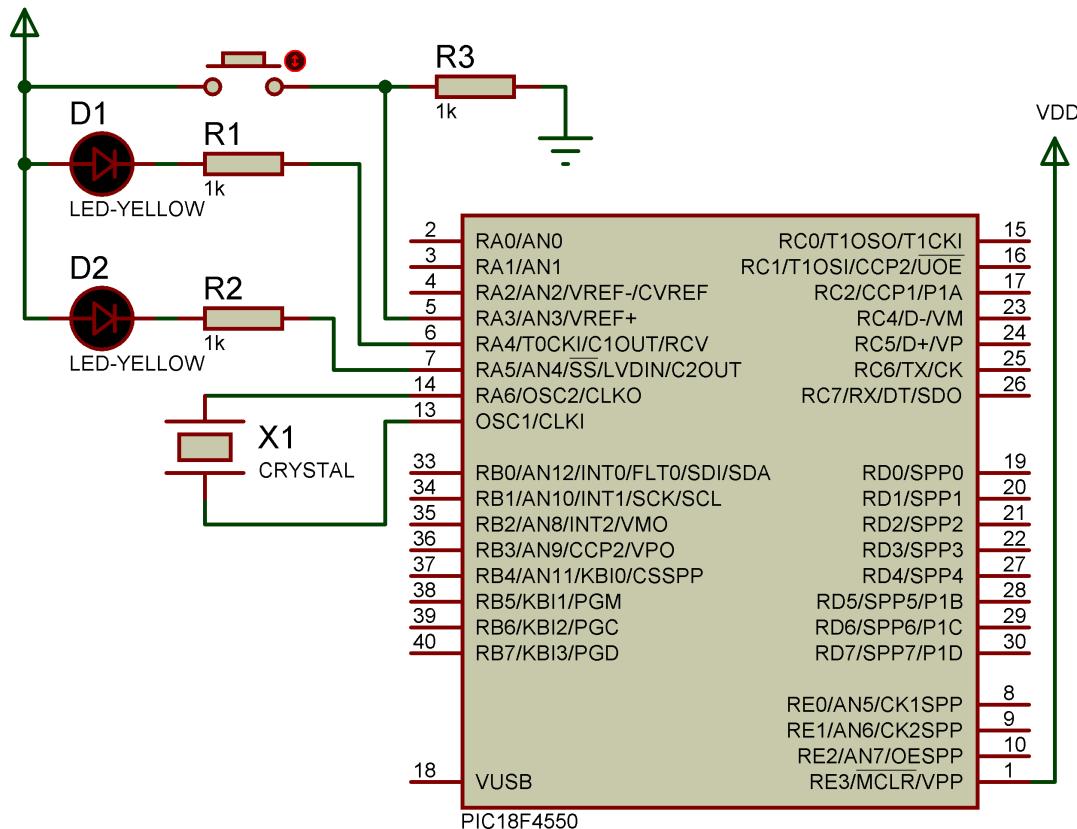
- Realizamos a instalação do ProtoN Compiler¹, e seguimos alguns passos do manual de utilização²;

¹ ProtoN Compiler: <<https://github.com/sftom/ee1054/ide/311-InstallPROTON.exe>>

² Manual do ProtoN Compiler: <<https://github.com/sftom/ee1054/ide/311-ManualPROTON.pdf>>

Figura 10: Diagrama Esquemático desenvolvido para o PIC18F4550 durante a Atividade 3.

Fonte: Produzido pelos autores.



- Em seguida, realizamos a instalação do MikroC Pro PIC 2014³ e manual de utilização⁴;
- Como pedido na atividade foi utilizado o MicroC e a PROTON;

A gravação dos códigos desenvolvidos no PIC18F4550 é realizada através de um gravador, como mostrado na Figura 14.

Para fonte de consulta, inserimos o Diagrama de Bloco, como pode ser observado na Figura 15, extraída do Data Sheet do PIC18F4550. (TECHNOLOGY, 2009).

³ MikroC Pro PIC 2014<https://github.com/sftom/ee1054/ide/312-mikroC_PRO_PIC_2014_Build.6.4.0.exe>

⁴ Manual do MikroC Pro PIC 2014<<https://github.com/sftom/ee1054/ide/312-ManualMikroc-pro-pic.pdf>>

Figura 15: Diagrama de Bloco do Data Sheet do PIC18F4550.

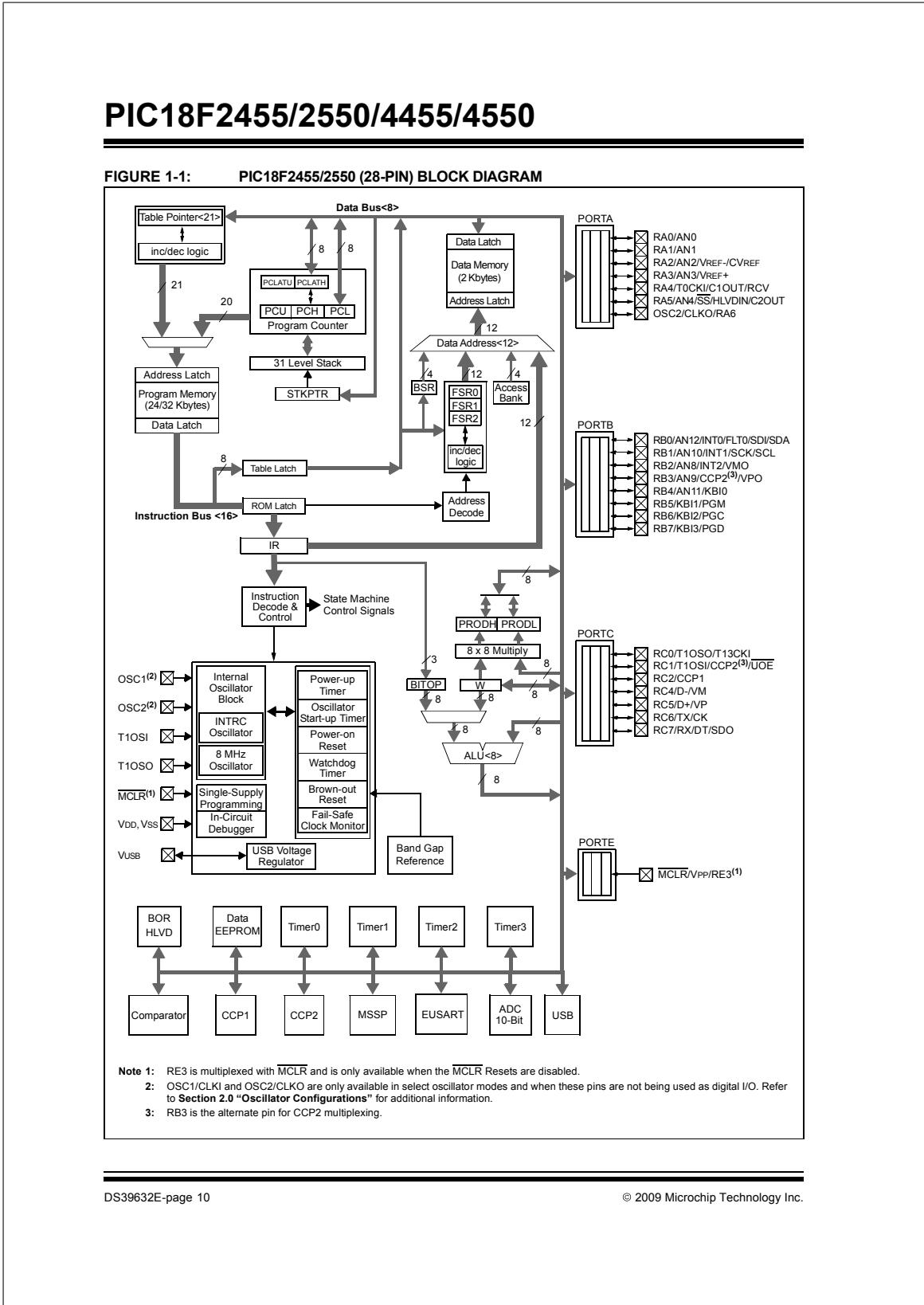


Figura 11: Programação em IDE MikroC.

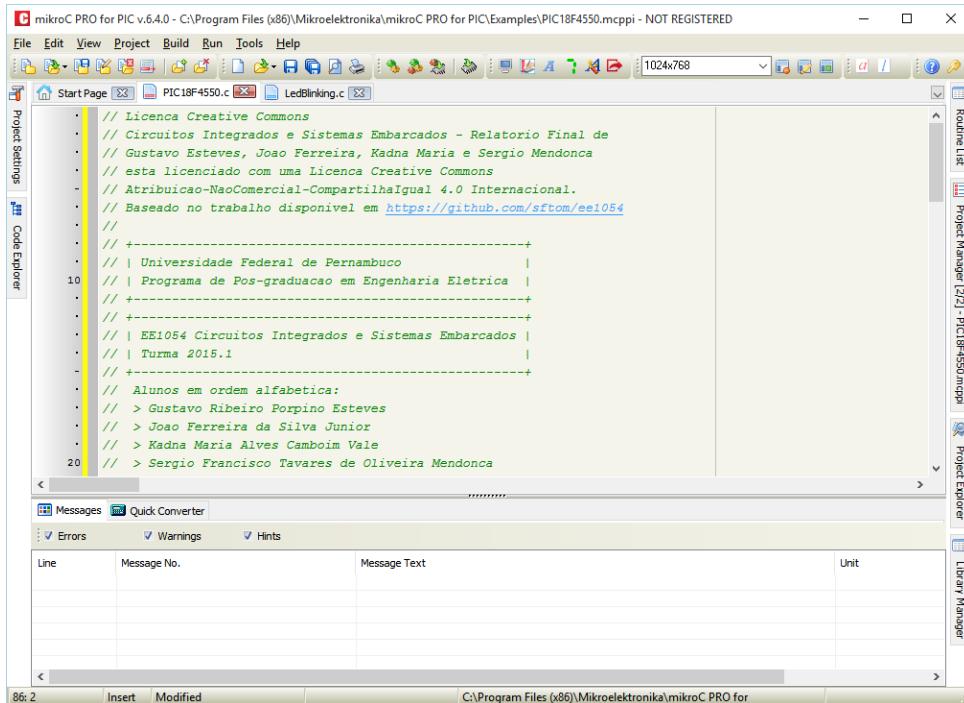
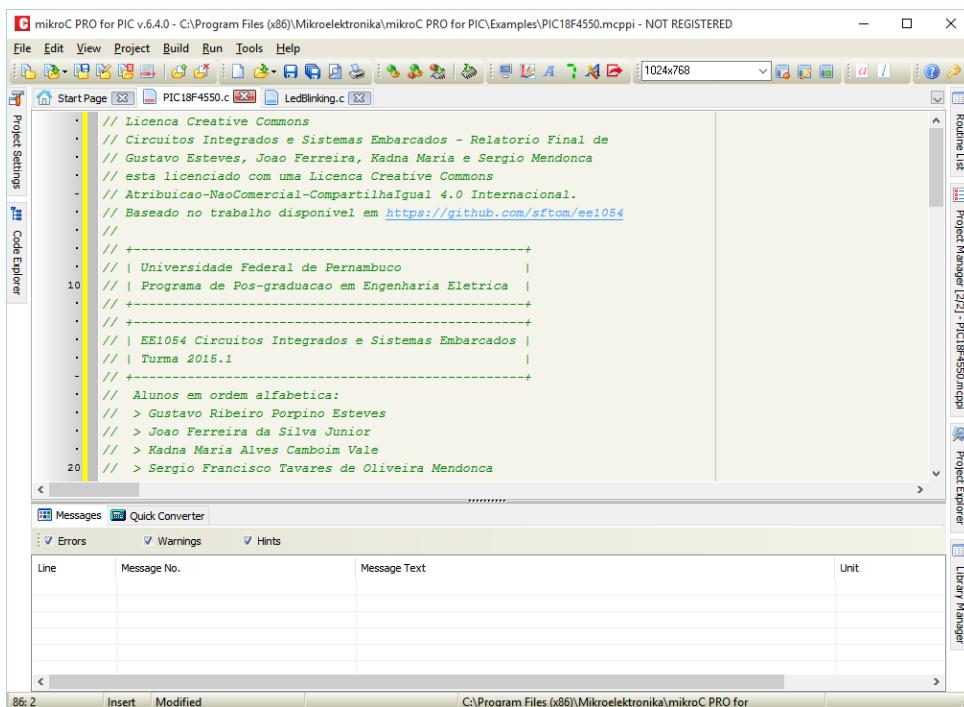


Figura 12: Configuração do microcontrolador na IDE MikroC.



Os códigos-fontes elaborados estão disponibilizados a seguir:

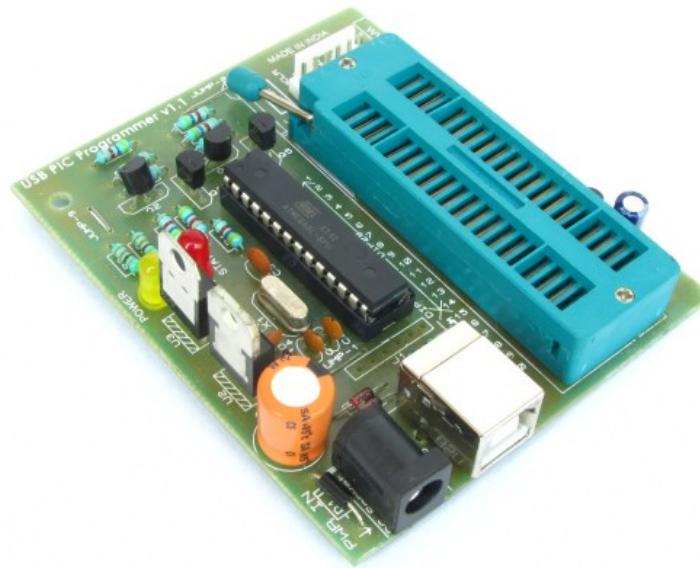
Microcontrolador PIC18F4550 – A03 (Códigos-fontes)

A03Q02

Figura 13: Configuração do Bootloader na aplicação PROTO’N.



Figura 14: Gravador de firmware para Microcontrolador PIC18F4550.



```

1 // Licenca Creative Commons
2 // Circuitos Integrados e Sistemas Embarcados - Relatorio Final de
3 // Gustavo Esteves, Joao Ferreira, Kadna Maria e Sergio Mendonca
4 // esta licenciado com uma Licenca Creative Commons
5 // Atribuicao-NaoComercial-CompartilhaIgual 4.0 Internacional.
6 // Baseado no trabalho disponivel em https://github.com/sftom/ee1054
7 //
8 // +-----+
9 // | Universidade Federal de Pernambuco | 
10 // | Programa de Pos-graduacao em Engenharia Eletrica | 
11 // +-----+

```

```
12 // +-----+
13 // | EE1054 Circuitos Integrados e Sistemas Embarcados |
14 // | Turma 2015.1 |
15 // +-----+
16 // Alunos em ordem alfabetica:
17 // > Gustavo Ribeiro Porpino Esteves
18 // > Joao Ferreira da Silva Junior
19 // > Kadna Maria Alves Camboim Vale
20 // > Sergio Francisco Tavares de Oliveira Mendonca
21 // +-----+
22 */
23 // Fazer um programa para acionar dois LEDs da placa PIC fornecida.
24 // Utilizar o arquivo "2-Manual Proton" para identificar os LEDs e
25 // e onde estao conectados.
26 */
27 #pragma orgall 0x1000
28
29 int i;
30 int x = 0;
31
32 void Board_Init(void);
33
34 void main(void) org 0x1000 {
35     Board_Init(); // inicia placa
36     // coloca so um dos LEDs aceso pra alternar
37     PORTA.RA5 = 0;
38     x = 0;
39     while (1) {
40         while (x == 1) {
41             // alterna estado dos LEDs
42             PORTA.RA4 ^= 1;
43             PORTA.RA5 ^= 1;
44             Delay_ms(100);
45             if (PORTA.RA3 == 0) {
46                 x = 0;
47                 while (PORTA.RA3 == 0) {}
48             }
49         }
50         while (x == 0) {
51             // alterna estado dos LEDs
52             PORTA.RA5 = 0;
53             for (i=0; i<10;i++) {
54                 PORTA.RA4 ^= 1;
55                 Delay_ms(100);
56                 if (PORTA.RA3 == 0) {
57                     x = 1;
58                     while (PORTA.RA3 == 0) {}
```

```

59         break;
60     }
61 }
62 PORTA.RA4 = 1;
63 PORTA.RA5 = 1;
64 Delay_ms(100);
65 PORTA.RA4 = 0;
66 PORTA.RA5 = 0;
67 Delay_ms(100);
68 }
69 }
70 }
71
72 void Board_Init (void) {
73     RCON = 0X80;          // LIMPA REGISTRO DE RESET
74     ADCON1 = 0x0F;        // CONFIGURA TODAS AS PORTAS ANALOGICAS
75                         // COMO I/O menos A0 (sensor temp/trimpot)
76     CMCON = 0x0F;        // (DESABILITAR COMPARADOR)
77     LATA = 0;
78     TRISA = 0b11001111;
79     T0CON = 0b11000101; // timer ativo, 8 bits,clock interno,
80                         // preescale 1:64
81     LATB = 0;            // LIMPA O LATCH DA PORTA B
82     TRISB = 0;            // tudo Saida
83     LATD = 0;
84     TRISD = 0x00;        // PORTA D TUDO SAIDA
85     LATE = 0;
86 }
```

sources/EE1054-Atividade03-Questao02-Led.c

A03Q03

```

1 // Licenca Creative Commons
2 // Circuitos Integrados e Sistemas Embarcados - Relatorio Final de
3 // Gustavo Esteves, Joao Ferreira, Kadna Maria e Sergio Mendonca
4 // esta licenciado com uma Licenca Creative Commons
5 // Atribuicao-NaoComercial-CompartilhaIgual 4.0 Internacional.
6 // Baseado no trabalho disponivel em https://github.com/sftom/ee1054
7 //
8 // +-----+
9 // | Universidade Federal de Pernambuco           |
10 // | Programa de Pos-graduacao em Engenharia Eletrica |
11 // +-----+
12 // +-----+
13 // | EE1054 Circuitos Integrados e Sistemas Embarcados |
14 // | Turma 2015.1                                     |
```

```
15 // +-----+
16 // Alunos em ordem alfabetica:
17 // > Gustavo Ribeiro Porpino Esteves
18 // > Joao Ferreira da Silva Junior
19 // > Kadna Maria Alves Camboim Vale
20 // > Sergio Francisco Tavares de Oliveira Mendonca
21 // +-----+
22 */
23 // Ao pressionar pela segunda vez a tecla fazer com que os 2 LEDs se
24 // acendam da seguinte forma:
25 // a) LED 1 acende e apaga a cada 100ms.
26 // b) Apos 10 piscadas do LED1 o LED2 fica acesso por 100ms.
27 */
28 #pragma orgall 0x1000
29
30 void Board_Init (void);
31 void Delay_LED (void);
32
33 void main (void) org 0x1000 {
34     Board_Init(); // inicia placa
35
36     PORTA = 1;      // acende todos - teste
37     PORTB = 1;
38     PORTC = 1;
39     PORTD = 1;
40     PORTE = 1;
41
42     Delay_LED ();
43
44     PORTA = 0;      // apaga todos
45     PORTB = 0;
46     PORTC = 0;
47     PORTD = 0;
48     PORTE = 0;
49
50     Delay_LED ();
51
52     while (1) {
53         PORTA.RA0 = 1; // Ativa LED
54         Delay_LED (); // Aguarda tempo aceso
55         PORTA.RA0 = 0; // apaga LED
56         PORTA.RA1 = 1; // e repete ...
57         PORTA.RA0 = 0; // apaga LED
58         Delay_LED ();
59         PORTA.RA1 = 0;
60         PORTA.RA2 = 1; // e repete ...
61         Delay_LED ();
```

```
62 PORTA.RA2 = 0;
63 PORTA.RA3 = 1; // e repete ...
64 Delay_LED ();
65 PORTA.RA3 = 0;
66 PORTA.RA4 = 1; // e repete ...
67 Delay_LED ();
68 PORTA.RA4 = 0;
69 PORTA.RA5 = 1; // e repete ...
70 Delay_LED ();
71 PORTA.RA5 = 0;
72 PORTE.RE0 = 1; // e repete ... PORT E
73 Delay_LED ();
74 PORTE.RE0 = 0;
75 PORTE.RE1 = 1; // e repete ...
76 Delay_LED ();
77 PORTE.RE1 = 0;
78 PORTE.RE2 = 1; // e repete ...
79 Delay_LED ();
80 PORTE.RE2 = 0;
81 PORTC.RC0 = 1; // e repete ... PORT C
82 Delay_LED ();
83 PORTC.RC0 = 0;
84 PORTC.RC1 = 1; // e repete ...
85 Delay_LED ();
86 PORTC.RC1 = 0;
87 PORTC.RC2 = 1; // e repete ...
88 Delay_LED ();
89 PORTC.RC2 = 0;
90 PORTC.RC6 = 1; // e repete ...
91 Delay_LED ();
92 PORTC.RC6 = 0;
93 PORTC.RC7 = 1; // e repete ...
94 Delay_LED ();
95 PORTC.RC7 = 0;
96 PORTD.RD0 = 1; // e repete ... PORT D
97 Delay_LED ();
98 PORTD.RD0 = 0;
99 PORTD.RD1 = 1; // e repete ... PORT D
100 Delay_LED ();
101 PORTD.RD1 = 0;
102 PORTD.RD2 = 1; // e repete ... PORT D
103 Delay_LED ();
104 PORTD.RD2 = 0;
105 PORTD.RD3 = 1; // e repete ... PORT D
106 Delay_LED ();
107 PORTD.RD3 = 0;
108 PORTD.RD4 = 1; // e repete ... PORT D
```

```
109     Delay_LED ();
110     PORTD.RD4 = 0;
111     PORTD.RD5 = 1; // e repete ... PORT D
112     Delay_LED ();
113     PORTD.RD5 = 0;
114     PORTD.RD6 = 1; // e repete ... PORT D
115     Delay_LED ();
116     PORTD.RD6 = 0;
117     PORTD.RD7 = 1; // e repete ... PORT D
118     Delay_LED ();
119     PORTD.RD7 = 0;
120     PORTB.RB0 = 1; // e repete ... PORT B
121     Delay_LED ();
122     PORTB.RB0 = 0;
123     PORTB.RB1 = 1; // e repete ... PORT B
124     Delay_LED ();
125     PORTB.RB1 = 0;
126     PORTB.RB2 = 1; // e repete ... PORT B
127     Delay_LED ();
128     PORTB.RB2 = 0;
129     PORTB.RB3 = 1; // e repete ... PORT B
130     Delay_LED ();
131     PORTB.RB3 = 0;
132     PORTB.RB4 = 1; // e repete ... PORT B
133     Delay_LED ();
134     PORTB.RB4 = 0;
135     PORTB.RB5 = 1; // e repete ... PORT B
136     Delay_LED ();
137     PORTB.RB5 = 0;
138     PORTB.RB6 = 1; // e repete ... PORT B
139     Delay_LED ();
140     PORTB.RB6 = 0;
141     PORTB.RB7 = 1; // e repete ... PORT B
142     Delay_LED ();
143     PORTB.RB7 = 0;
144 }
145 }
146
147 void Delay_LED (void) {
148     Delay_ms (40);
149 }
150
151 void Board_Init (void) {
152     RCON = 0X80; // LIMPA REGISTRO DE RESET
153     ADCON1 = 0x0F; // CONFIGURA TODAS AS PORTAS ANALOGICAS COMO I/O
154     CMCON = 0x0F; // DESABILITA COMPARADOR
155     T0CON = 0b11000101; // timer ativo, 8 bits,clock interno
```

```
156 TRISA = 0;  
157 TRISB = 0; // todos Saida  
158 TRISC = 0;  
159 TRISD = 0;  
160 TRISE = 0;  
161 }
```

sources/EE1054-Atividade03-Questao03-Leds-Sequencial.c

4 Microcontrolador MSP430

4.1 Atividades propostas

Relatório de experimentos – Aula prática com microcontrolador família MSP430

Atividade 4

1. Instalar o compilador Code Composer Studio (CCS) Integrated Development Environment (IDE¹) for MSP Microcontrollers.
2. Realizar testes para piscar LEDs com a placa LAUNCHPAD MSP-exp430G2 e também com a placa eZ430-RF2500T.
3. Fazer um programa para ler um sinal proveniente de dois botões externos a placa e realizar o acionamento de 4 LEDs, conforme a tabela abaixo (escolher qualquer uma das placas), conforme a Tabela 1:

Tabela 1: Esquema de programação da da MSP430.

Botão 1	Botão 2	Acionamento do LED
1	0	LED1
0	1	LED2
1	1	LED3
0	0	LED4

Os LEDs deverão acender após a combinação realizada e permanecer durante 1 segundo aceso.

4. Fazer um programa para gerar pulsos em um pino de saída, bem como acender um indicador (LED) ao mesmo tempo. Utilizar duas teclas externas uma para aumentar a velocidade das piscadas do LED e outra para diminuir a velocidade das piscadas do LED. Pede-se o controle de velocidade de 1 a 60 piscadas por minuto.

4.2 Relatório da Atividade

Para fonte de consulta, inserimos o Diagrama de Bloco, como pode ser observado na Figura 20, extraída do Data Sheet do MSP430. ([INSTRUMENTS, 2012](#)).

¹ <http://www.ti.com/tool/ccstudio-msp>

Figura 20: Diagrama de Bloco do Data Sheet do MSP430.

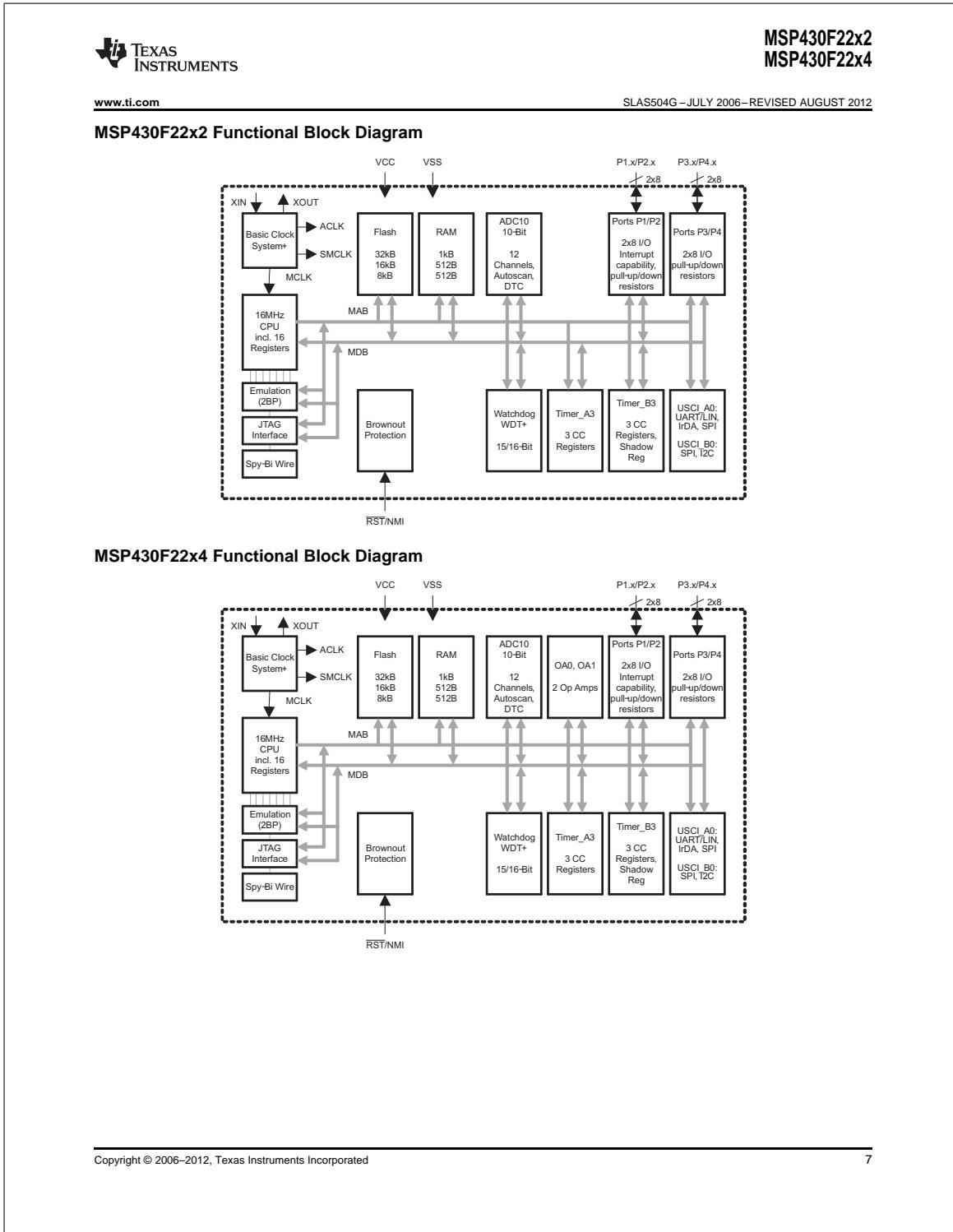


Figura 16: Microcontrolador EZ430-RF2500T e seu Gravador de Firmware.

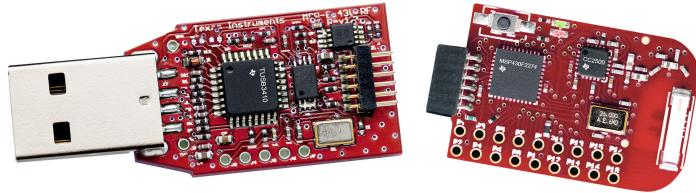
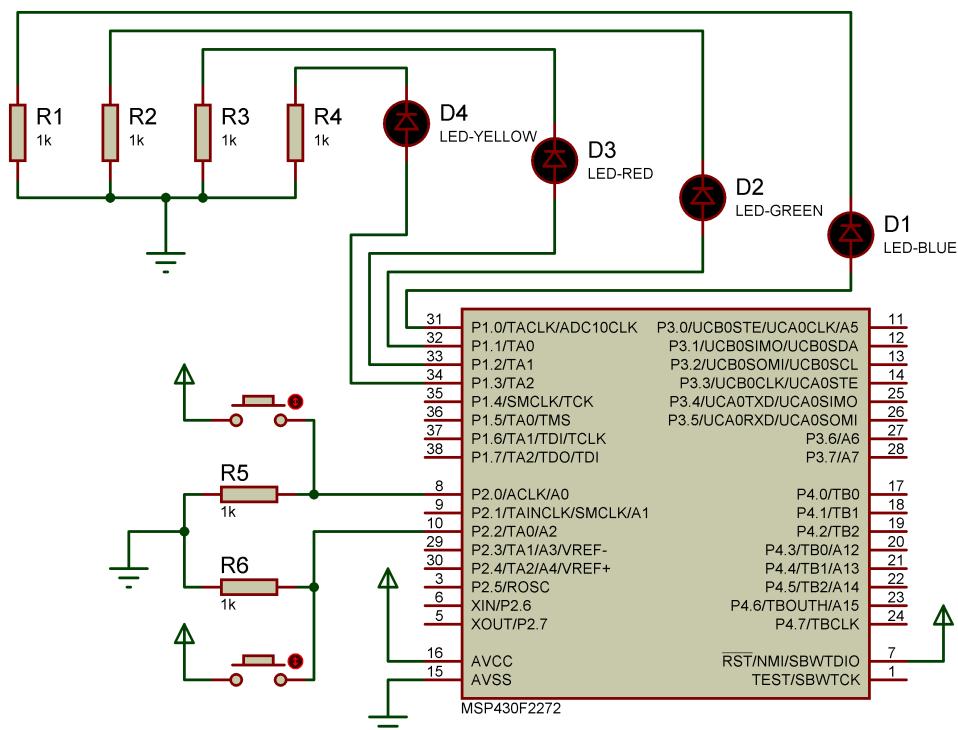


Figura 17: Diagrama Esquemático desenvolvido para o MSP430F2272 durante a Atividade 4.

Fonte: Produzido pelos autores.



Os códigos-fontes elaborados estão disponibilizados a seguir:

Microcontrolador MSP430 – A04 (Códigos-fontes)

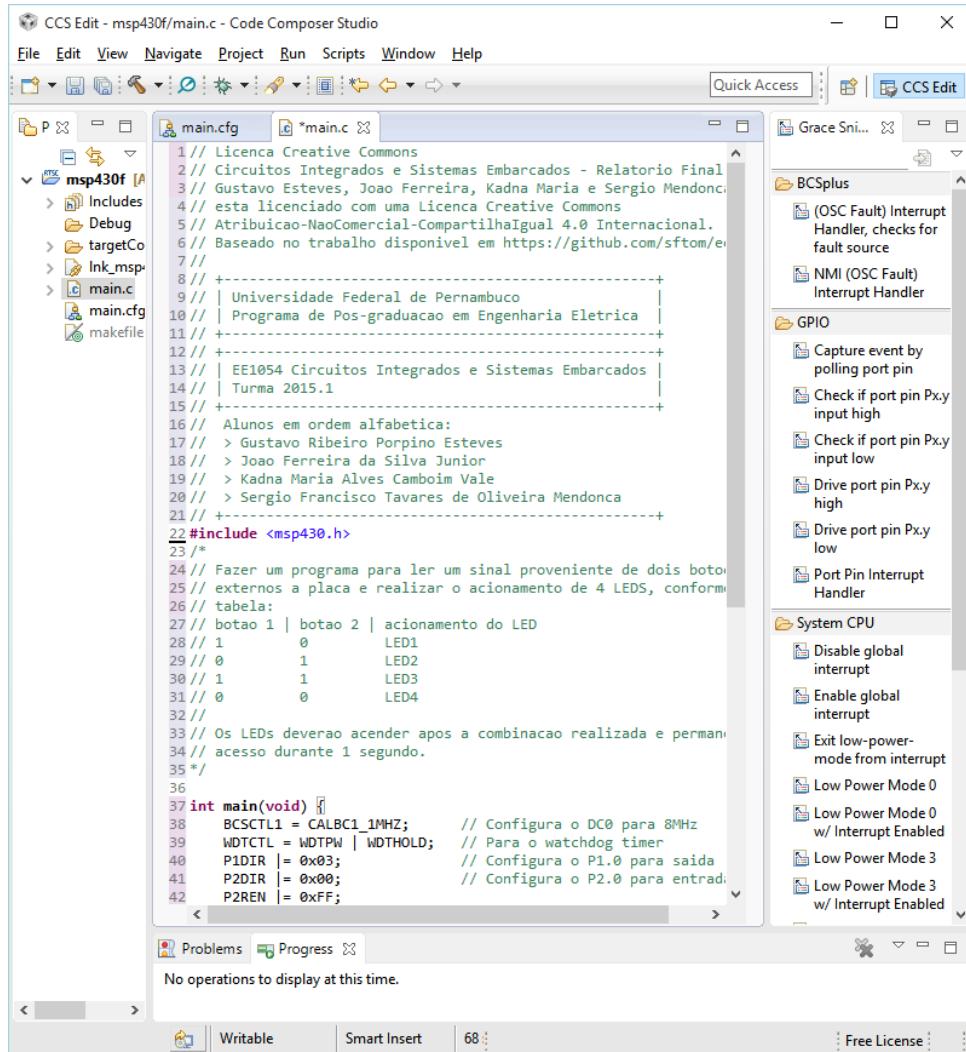
A04Q04

```

1 // Licenca Creative Commons
2 // Circuitos Integrados e Sistemas Embarcados - Relatorio Final de
3 // Gustavo Esteves, Joao Ferreira, Kadna Maria e Sergio Mendonca
4 // esta licenciado com uma Licenca Creative Commons
5 // Atribuicao-NaoComercial-CompartilhaIgual 4.0 Internacional.
6 // Baseado no trabalho disponivel em https://github.com/sftom/ee1054
7 //
8 // +-----+

```

Figura 18: Programação do microcontrolador MSP430F2272 na IDE CodeComposer.

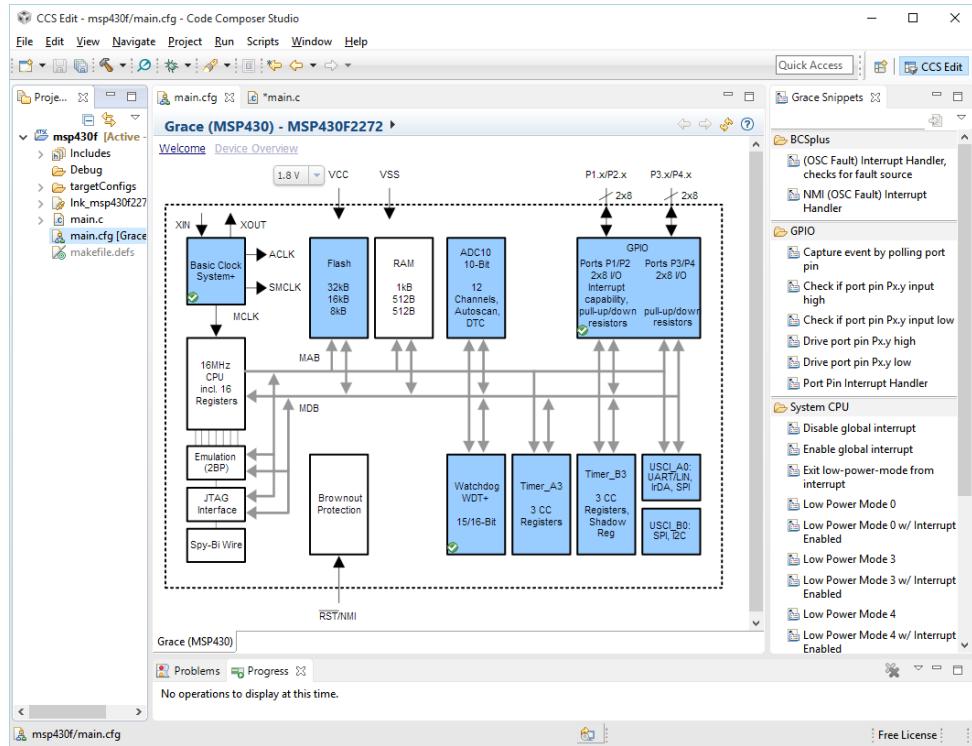


```

9 // | Universidade Federal de Pernambuco |
10 // | Programa de Pos-graduacao em Engenharia Eletrica |
11 // +-----+
12 // +-----+
13 // | EE1054 Circuitos Integrados e Sistemas Embarcados |
14 // | Turma 2015.1 |
15 // +-----+
16 // | Alunos em ordem alfabetica: |
17 // > Gustavo Ribeiro Porpino Esteves |
18 // > Joao Ferreira da Silva Junior |
19 // > Kadna Maria Alves Camboim Vale |
20 // > Sergio Francisco Tavares de Oliveira Mendonca |
21 // +-----+
22 #include <msp430.h>
23 /*
24 // Fazer um programa para gerar pulsos em um pino de saida, bem como
25 // acender um indicador (LED) ao mesmo tempo. Utilizar duas teclas
26 // externas uma para aumentar a velocidade das piscadas do LED e outra

```

Figura 19: Programação do microcontrolador MSP430F2272 na IDE CodeComposer.



```

27 // para diminuir a velocidade das piscadas do LED. Pede-se o controle
28 // de velocidade de 1 a 60 piscadas por minuto.
29 */
30
31 int main(void) {
32     BCSCTL1 = CALBC1_1MHZ;      // Configura o DC0 para 8MHz
33     WDTCTL = WDTPW | WDTHOLD; // Para o watchdog timer
34     P1DIR |= 0x03;           // Configura o P1.0 para saída
35     P2DIR |= 0x00;           // Configura o P2.0 para entrada
36     P2REN |= 0xFF;
37     P3DIR |= 0x03;           // Configura o P3.0 para saída
38     long int g = 5;
39     for(;;) {
40         volatile unsigned long int i;
41         if ((P2IN & BIT0) == 1) {
42             g++;
43             if (g>60) g = 60;
44         }
45         if ((P2IN & BIT2) == 4) {
46             g--;
47             if (g<1) g = 1;
48         }
49
50         P1OUT = 0x01;
51         i = g*10000;
52         do i--;

```

```
53     while(i != 0);  
54  
55     if ((P2IN & BIT0) == 1) {  
56         g++;  
57         if (g>60) g = 60;  
58     }  
59     if ((P2IN & BIT2) == 4) {  
60         g--;  
61         if (g<1) g = 1;  
62     }  
63  
64     P1OUT = 0x00;  
65     i = g*10000;  
66     do i--;  
67     while(i != 0);  
68 }  
69 return 0;  
70 }
```

sources/EE1054-Atividade04-Questao04-Pulso.c

5 Microcontrolador MSP430 – GRACE

5.1 Atividades propostas

Relatório de Experimentos – Aula prática com MSP430 – GRACE

Atividade 5

1. Instalar a ferramenta de desenvolvimento GRACE para a família MSP430 (disponível do site da Texas).
2. Escolher um microcontrolador da família MSP430 que possua pelo menos um conversor A/D de 10 bits, interface de I/O para periféricos, memória do tipo flash e comunicação serial assíncrona do tipo RS232. Criar um software compatível no CCS, utilizando o GRACE, com as seguintes características:
 - a) Configurar o conversor A/D para realizar a aquisição de dois canais, com uma taxa de 100 amostras por segundo.
 - b) Realizar a conversão de 100 amostras, em cada canal.
 - c) Gravar as 100 amostras, de cada canal em uma memória FLASH.
 - d) Recuperar as 100 amostras, por canal, gravadas na memória FLASH e enviar via comunicação serial, padrão RS232 a uma velocidade de 9600 bauds, para um periférico externo.
3. Apresentar um programa completo com as especificações do item 2, com os referidos comentários de como foi realizado todo o processo. Dica: gerar os códigos com o GRACE e posteriormente chamá-los com uma função através do programa principal, construído no CCS.

5.2 Relatório da Atividade

Para fonte de consulta, inserimos o Diagrama de Bloco, como pode ser observado na Figura 25, extraída do Data Sheet do MSP430F2272 (GRACE). ([INSTRUMENTS, 2012](#)).

Figura 25: Diagrama de Bloco do Data Sheet do MSP430.

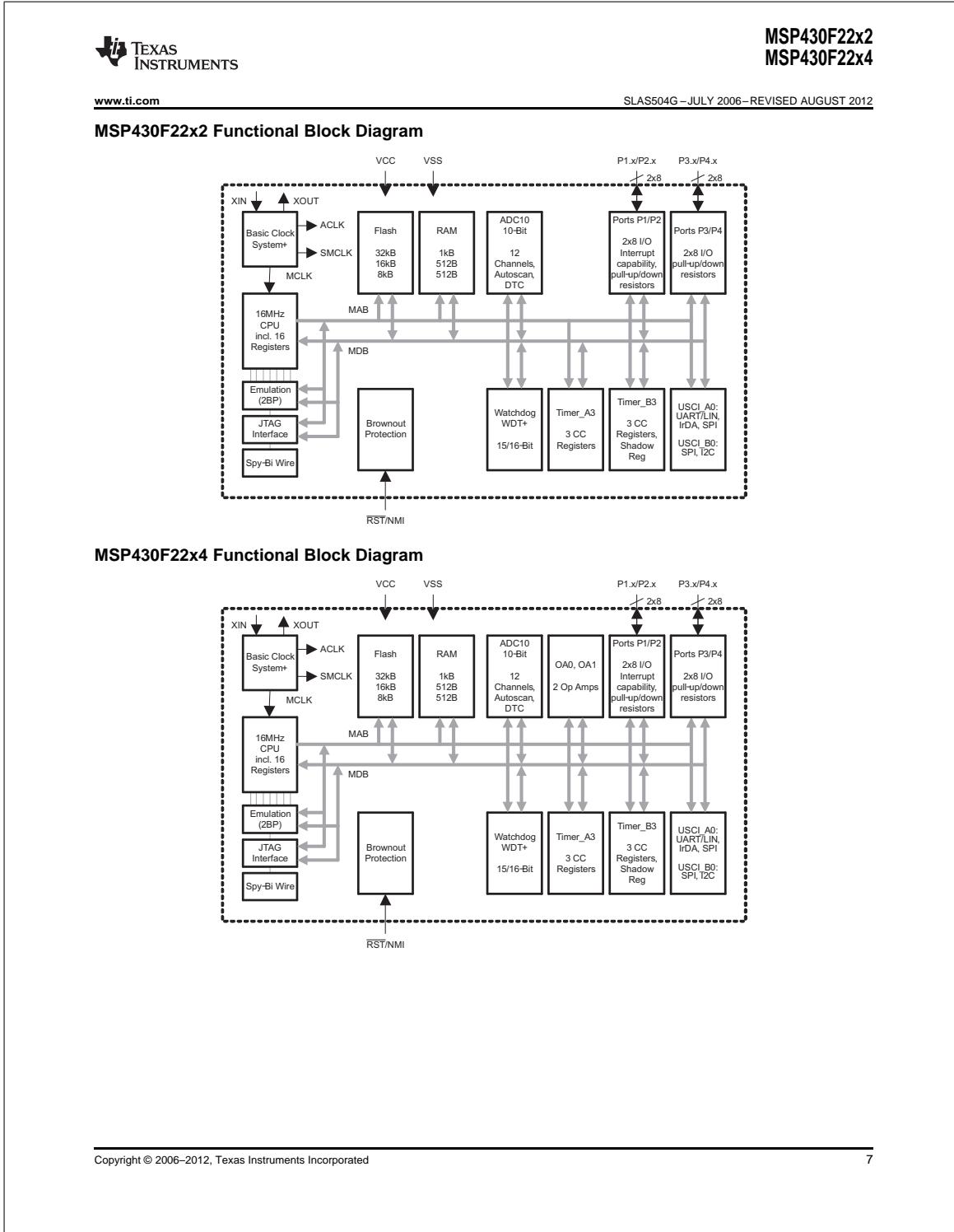
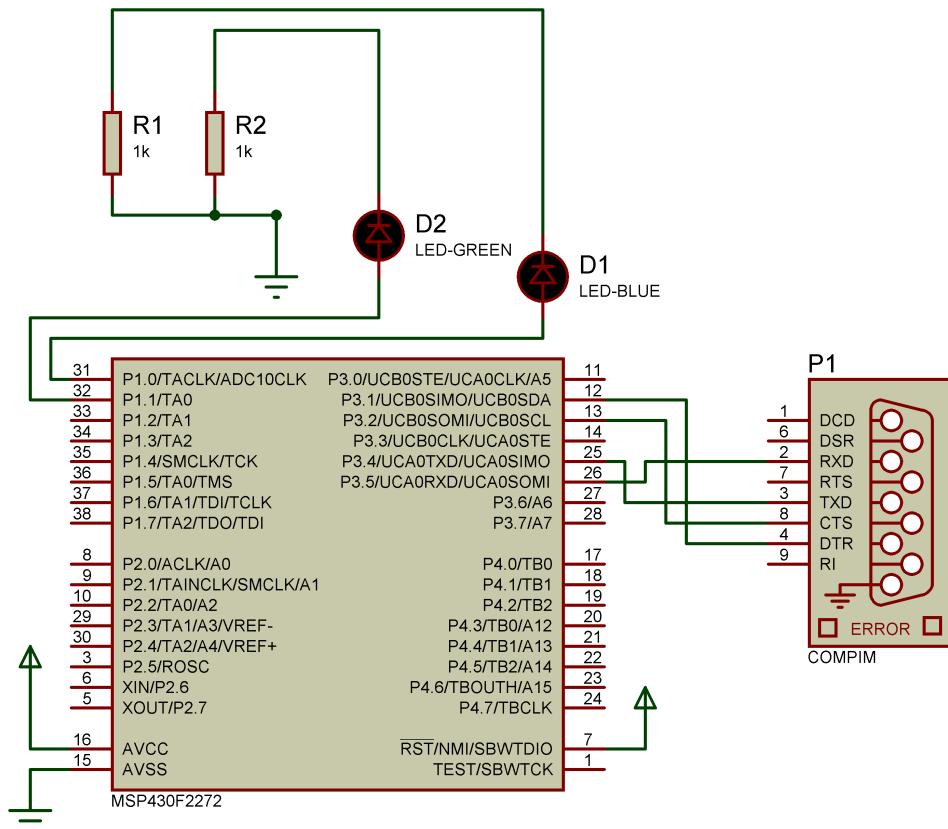


Figura 21: Diagrama Esquemático desenvolvido para o MSP430F2272 (GRACE) durante a Atividade 5.

Fonte: Produzido pelos autores.



Os códigos-fontes elaborados estão disponibilizados a seguir:

MSP430 – Grace – A05 (Códigos-fontes)

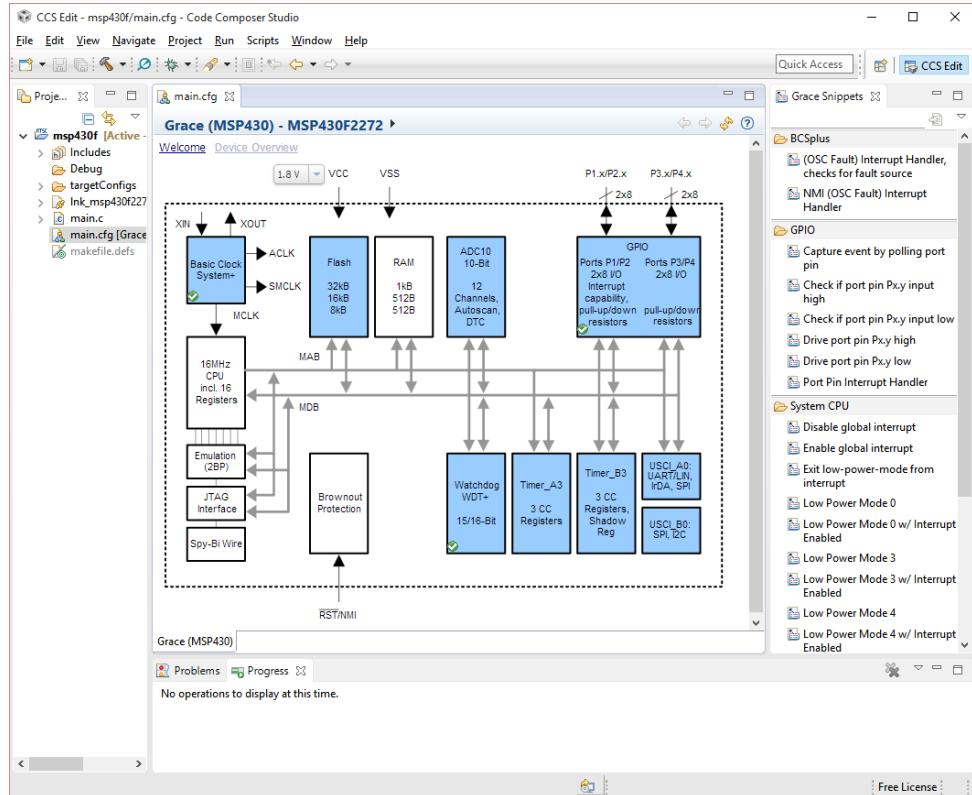
A05Q02-03

```

1 // Licenca Creative Commons
2 // Circuitos Integrados e Sistemas Embarcados - Relatorio Final de
3 // Gustavo Esteves, Joao Ferreira, Kadna Maria e Sergio Mendonca
4 // esta licenciado com uma Licenca Creative Commons
5 // Atribuicao-NaoComercial-CompartilhaIgual 4.0 Internacional.
6 // Baseado no trabalho disponivel em https://github.com/sftom/ee1054
7 //
8 // +-----+
9 // | Universidade Federal de Pernambuco | |
10 // | Programa de Pos-graduacao em Engenharia Eletrica | |
11 // +-----+
12 // +-----+
13 // | EE1054 Circuitos Integrados e Sistemas Embarcados | |
14 // | Turma 2015.1 | |

```

Figura 22: Programação da MSP430F2272 na IDE GRACE (ambiente de simulação).

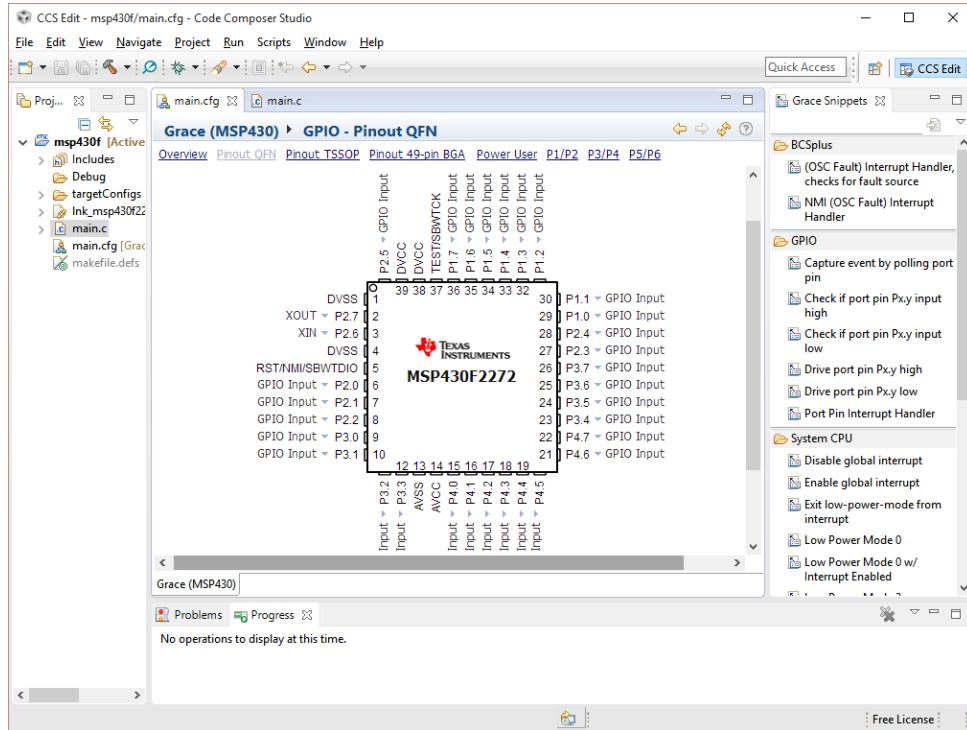


```

15 // +-----+
16 // Alunos em ordem alfabetica:
17 // > Gustavo Ribeiro Porpino Esteves
18 // > Joao Ferreira da Silva Junior
19 // > Kadna Maria Alves Camboim Vale
20 // > Sergio Francisco Tavares de Oliveira Mendonca
21 // +-----+
22 #include <msp430f2274.h>
23 #include <stdlib.h>
24 #include <stdio.h>
25 #include <string.h>
26 #include <math.h>
27
28 void swap(int* a, int* b);
29 void flash_write(unsigned int valor[100]);
30 int flash_read(int posicao);
31 char* itoa(int num, char* str, int base);
32 extern void BCSplus_init();
33 extern void USCI_A0_init();
34 volatile int a;
35 char array[5];
36 unsigned int var;
37 volatile int contador=0;
38 unsigned int amostras[100];

```

Figura 23: Programação da MSP430F2272 na IDE GRACE (ambiente de simulação).

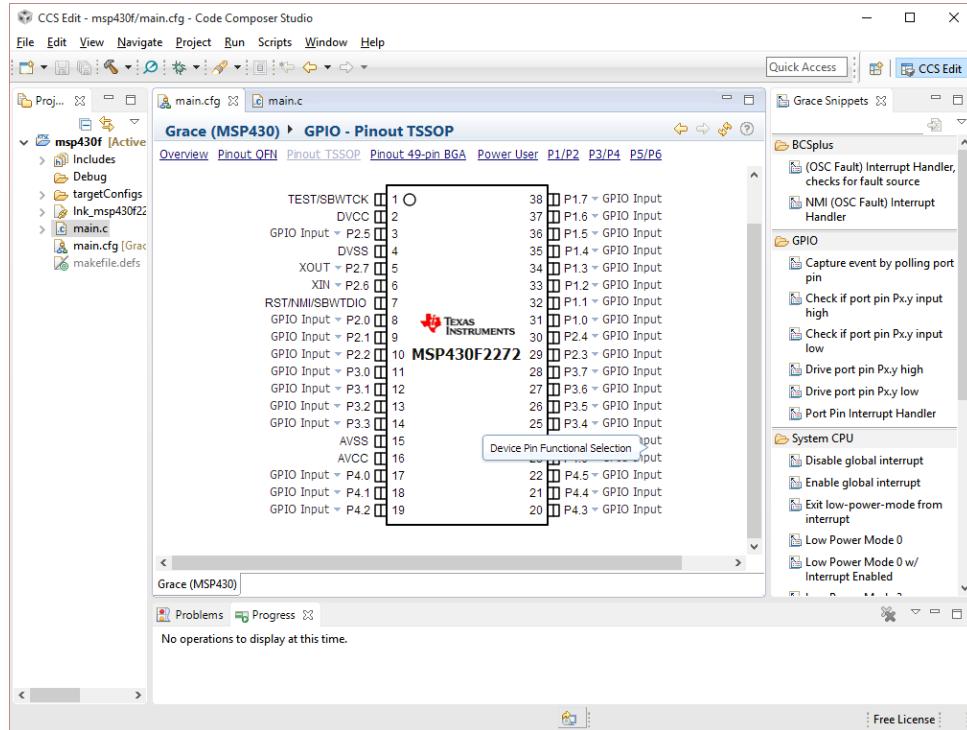


```

39 int k;
40
41 int main(void) {
42     WDTCTL = WDTPW | WDTHOLD; // Stop watchdog timer
43     unsigned int j;
44     WDTCTL = WDTPW + WDTHOLD; // Stop watchdog timer
45     if (CALBC1_1MHZ == 0xFF || CALDCO_1MHZ == 0xFF) {
46         while(1); // If calibration constants erased
47             // do not load, trap CPU!!
48     }
49     BCSCTL1 = CALBC1_1MHZ; // Set DCO to 1MHz
50     BCOCTL = CALDCO_1MHZ;
51     FCTL2 = FWKEY + FSSEL0 + FN1; // MCLK/3 for Flash Timing Generator
52     //configuracao da serial
53     BCSplus_init();
54     P3SEL = 0x30; // P3.4,5 = USCI_A0 TXD/RXD
55     USCI_A0_init();
56     P1DIR |= 0x03; // COLOCA P1.0 E P1.1 COMO SAIDAS
57             // (leds 1 (vermelho) e 2 (verde))
58     P1OUT = 0x00;
59     IE2 |= UCA0RXIE; // Enable USCI_A0 RX interrupt
60     ADC10CTL1 = CONSEQ_2; // Repeat single channel
61     ADC10CTL0 = ADC10SHT_2 + MSC + ADC10ON + ADC10IE;
62     ADC10DTC1 = 0x64; // 100 conversions
63     ADC10AE0 |= 0x03; // P2.0 e P2.1 ADC option select
64     P1DIR |= 0x01; // Set P1.0 to output direction

```

Figura 24: Programação da MSP430F2272 na IDE GRACE (ambiente de simulação).



```

65     __delay_cycles(4000000);
66 //loop infinito
67 while (1) {
68     while (contador<100) {
69         ADC10CTL0 &= ~ENC;
70         while (ADC10CTL1 & BUSY); // Wait if ADC10 core is active
71         ADC10SA = 0x200; // Data buffer start
72         P1OUT |= 0x01; // Set P1.0 LED on
73         ADC10CTL0 |= ENC + ADC10SC; // Sampling and conversion start
74         __bis_SR_register(CPUOFF + GIE);
75     }
76     contador=0;
77     ADC10CTL0 &= ~ENC;
78     flash_write(amostras);
79     P1OUT ^= 0x01;
80     __delay_cycles(2000000);
81     _DINT();
82     for(j=0;j<100;j++) {
83         // envio continuo para serial
84         while (!(IFG2 & UCA0TXIFG)); // USCI_A0 TX buffer ready?
85         UCA0TXBUF = 'x';
86         var = flash_read(j);
87         strcpy(array, itoa(var,array,10));
88         while (!(IFG2 & UCA0TXIFG)); // USCI_A0 TX buffer ready?
89         UCA0TXBUF =array[4];
90         __delay_cycles(3000);

```

```
91     while (!(IFG2 & UCA0TXIFG)); // USCI_A0 TX buffer ready?
92     UCA0TXBUF =array[3];
93     __delay_cycles(3000);
94     while (!(IFG2 & UCA0TXIFG)); // USCI_A0 TX buffer ready?
95     UCA0TXBUF =array[2];
96     __delay_cycles(3000);
97     while (!(IFG2 & UCA0TXIFG)); // USCI_A0 TX buffer ready?
98     UCA0TXBUF =array[1];
99     __delay_cycles(3000);
100    while (!(IFG2 & UCA0TXIFG)); // USCI_A0 TX buffer ready?
101    UCA0TXBUF =array[0];
102    __delay_cycles(3000);
103 }
104 }
105 }
106
107 void swap(int* a, int* b) {
108     int* temp = a;
109     a = b;
110     b = temp;
111 }
112
113 void reverse(char str[], int length) {
114     int start = 0;
115     int end = length -1;
116     while (start < end) {
117         swap(*(str+start), *(str+end));
118         start++;
119         end--;
120     }
121 }
122
123 //converte um inteiro de 16 bits em um array de 4 inteiros codificados em
124 //ascii
125 char* itoa(int num, char* str, int base) {
126     int i = 0;
127     int isNegative = 0;
128     /* Handle 0 explicitely, otherwise empty string is printed for 0 */
129     if (num == 0) {
130         str[i++] = '0';
131         str[i] = '\0';
132         return str;
133     }
134     // In standard itoa(), negative numbers are handled only with
135     // base 10. Otherwise numbers are considered unsigned.
136     if (num < 0 && base == 10) {
137         isNegative = 1;
```

```
137     num = -num;
138 }
139 // Process individual digits
140 while (num != 0) {
141     int rem = num % base;
142     str[i++] = (rem > 9)? (rem-10) + 'a' : rem + '0';
143     num = num/base;
144 }
145 // If number is negative, append '-'
146 if (isNegative) {
147     str[i++] = '-';
148 }
149 str[i] = '\0'; // Append string terminator
150 // Reverse the string
151 reverse(str, i);
152 return str;
153 }
154
155 void flash_write(unsigned int valor[100]) {
156     int i;
157     int *Flash_ptr; // Flash pointer
158     Flash_ptr = (int *)0x1000;
159     FCTL3 = FWKEY; // Clear Lock bit to enable erase access
160     FCTL1 = FWKEY+ERASE; // ERASE bit = 1; segment erase mode selected
161     *Flash_ptr = 0; // Dummy write to erase Flash segment
162     // Write to flash segment D
163     FCTL1 = FWKEY + WRT; // Set WRT bit for byte/word write operation
164     for (i = 0; i <200 ; i++) {
165         *Flash_ptr++ = valor[i]; // Byte write to flash location
166     }
167 }
168
169 int flash_read(int posicao) {
170     int valor;
171     int *Flash_ptr = (int *)0x1000;
172     *Flash_ptr=Flash_ptr+posicao;
173     valor = *Flash_ptr;
174     FCTL1 = FWKEY; // Clear WRT bit
175     FCTL3 = FWKEY + LOCK; // Set LOCK bit to lock flash memory
176     return valor;
177 }
178 // interrupcao do RX da porta serial
179 #pragma vector=USCIAB0RX_VECTOR
180 __interrupt void USCI0RX_ISR(void) {
181 }
182 // ADC10 interrupt service routine
183 #pragma vector=ADC10_VECTOR
```

```

184 __interrupt void ADC10_ISR(void) {
185     //interrupcao da conversao. Sale o valor da conversao aqui.
186     amostras[contador]=ADC10MEM;
187     contador++;
188     __bic_SR_register_on_exit(CPUOFF); // Clear CPUOFF bit from 0(SR)
189 }
```

sources/EE1054-Atividade05-adc_flash_serial.c

```

1 // Licenca Creative Commons
2 // Circuitos Integrados e Sistemas Embarcados - Relatorio Final de
3 // Gustavo Esteves, Joao Ferreira, Kadna Maria e Sergio Mendonca
4 // esta licenciado com uma Licenca Creative Commons
5 // Atribuicao-NaoComercial-CompartilhaIgual 4.0 Internacional.
6 // Baseado no trabalho disponivel em https://github.com/sftom/ee1054
7 //
8 // +-----+
9 // | Universidade Federal de Pernambuco           |
10 // | Programa de Pos-graduacao em Engenharia Eletrica |
11 // +-----+
12 // +-----+
13 // | EE1054 Circuitos Integrados e Sistemas Embarcados |
14 // | Turma 2015.1                                         |
15 // +-----+
16 // Alunos em ordem alfabetica:
17 // > Gustavo Ribeiro Porpino Esteves
18 // > Joao Ferreira da Silva Junior
19 // > Kadna Maria Alves Camboim Vale
20 // > Sergio Francisco Tavares de Oliveira Mendonca
21 // +-----+
22 /*
23 * This file is automatically generated and does not require a license
24 *
25 * ===== WARNING: CHANGES TO THIS GENERATED FILE WILL BE OVERWRITTEN =====
26 *
27 * To make changes to the generated code, use the space between existing
28 *      "USER CODE START (section: <name>)"
29 * and
30 *      "USER CODE END (section: <name>)"
31 * comments, where <name> is a single word identifying the section.
32 * Only these sections will be preserved.
33 *
34 * Do not move these sections within this file or change the START and
35 * END comments in any way.
36 * ===== ALL OTHER CHANGES WILL BE OVERWRITTEN WHEN IT IS REGENERATED =====
37 *
38 * This file was generated from
```

```
39 *      C:/ti/grace/grace_2_20_02_32/packages/ti/mcu/msp430/csl/clock/
40 *      BCSplus_init.xdt
41 */
42 #include <msp430f2274.h>
43 /* USER CODE START (section: BCSplus_init_c_prologue) */
44 /* User defined includes, defines, global variables and functions */
45 /* USER CODE END (section: BCSplus_init_c_prologue) */
46 /*
47 * ===== BCSplus_graceInit =====
48 * Initialize MSP430 Basic Clock System
49 */
50 void BCSplus_init(void) {
51     /* USER CODE START (section: BCSplus_graceInit_prologue) */
52     /* User initialization code */
53     /* USER CODE END (section: BCSplus_graceInit_prologue) */
54     /*
55     * Basic Clock System Control 2
56     *
57     * SELM_0 -- DCOCLK
58     * DIVM_0 -- Divide by 1
59     * ~SELS -- DCOCLK
60     * DIVS_0 -- Divide by 1
61     * ~DCOR -- DCO uses internal resistor
62     *
63     * Note: ~<BIT> indicates that <BIT> has value zero
64     */
65     BCSCTL2 = SELM_0 | DIVM_0 | DIVS_0;
66     if (CALBC1_1MHZ != 0xFF) {
67         /* Follow recommended flow. First, clear all DCOx and MODx bits. Then
68         * apply new RSELx values. Finally, apply new DCOx and MODx bit values.
69         */
70         DCOCTL = 0x00;
71         BCSCTL1 = CALBC1_1MHZ;          /* Set DCO to 1MHz */
72         DCOCTL = CALDCO_1MHZ;
73     }
74     /*
75     * Basic Clock System Control 1
76     *
77     * XT2OFF -- Disable XT2CLK
78     * ~XTS -- Low Frequency
79     * DIVA_0 -- Divide by 1
80     *
81     * Note: ~XTS indicates that XTS has value zero
82     */
83     BCSCTL1 |= XT2OFF | DIVA_0;
84     /*
```

```

85 * Basic Clock System Control 3
86 *
87 * XT2S_0 -- 0.4 - 1 MHz
88 * LFXT1S_0 -- If XTS = 0, XT1 = 32768kHz Crystal ; If XTS = 1, XT1 = 0.4
89 - 1-MHz crystal or resonator
90 * XCAP_1 -- ~6 pF
91 */
92 BCSCTL3 = XT2S_0 | LFXT1S_0 | XCAP_1;
93 /* USER CODE START (section: BCSplus_graceInit_epilogue) */
94 /* User code */
95 /* USER CODE END (section: BCSplus_graceInit_epilogue) */
96 }
```

sources/EE1054-Atividade05-BCSplus_init.c

```

1 // Licenca Creative Commons
2 // Circuitos Integrados e Sistemas Embarcados - Relatorio Final de
3 // Gustavo Esteves, Joao Ferreira, Kadna Maria e Sergio Mendonca
4 // esta licenciado com uma Licenca Creative Commons
5 // Atribuicao-NaoComercial-CompartilhaIgual 4.0 Internacional.
6 // Baseado no trabalho disponivel em https://github.com/sftom/ee1054
7 //
8 // +-----+
9 // | Universidade Federal de Pernambuco | |
10 // | Programa de Pos-graduacao em Engenharia Eletrica | |
11 // +-----+
12 // +-----+
13 // | EE1054 Circuitos Integrados e Sistemas Embarcados | |
14 // | Turma 2015.1 | |
15 // +-----+
16 // Alunos em ordem alfabetica:
17 // > Gustavo Ribeiro Porpino Esteves
18 // > Joao Ferreira da Silva Junior
19 // > Kadna Maria Alves Camboim Vale
20 // > Sergio Francisco Tavares de Oliveira Mendonca
21 // +-----+
22 /*
23 * This file is automatically generated and does not require a license
24 *
25 * ===== WARNING: CHANGES TO THIS GENERATED FILE WILL BE OVERWRITTEN =====
26 *
27 * To make changes to the generated code, use the space between existing
28 * "USER CODE START (section: <name>)"
29 * and
30 * "USER CODE END (section: <name>)"
31 * comments, where <name> is a single word identifying the section.
32 * Only these sections will be preserved.
33 *
```

```
34 * Do not move these sections within this file or change the START and
35 * END comments in any way.
36 * ===== ALL OTHER CHANGES WILL BE OVERWRITTEN WHEN IT IS REGENERATED =====
37 *
38 * This file was generated from
39 *      C:/ti/grace/grace_2_20_02_32/packages/ti/mcu/msp430/cs1/
40 *      communication/USCI_A0_init.xdt
41 */
42 #include <msp430f2274.h>
43 /* USER CODE START (section: USCI_A0_init_c_prologue) */
44 /* User defined includes, defines, global variables and functions */
45 /* USER CODE END (section: USCI_A0_init_c_prologue) */
46 /*
47 * ===== USCI_A0_graceInit =====
48 * Initialize Universal Serial Communication Interface A0 UART 2xx
49 */
50 void USCI_A0_init(void) {
51     /* USER CODE START (section: USCI_A0_graceInit_prologue) */
52     /* User initialization code */
53     /* USER CODE END (section: USCI_A0_graceInit_prologue) */
54     //IE2 |= UCA0RXIE;
55     /* Disable USCI */
56     UCA0CTL1 |= UCSWRST;
57     /*
58     * Control Register 1
59     *
60     * UCSSEL_2 -- SMCLK
61     * ~UCRXEIE -- Erroneous characters rejected and UCAxRXIFG is not set
62     * ~UCBRKIE -- Received break characters do not set UCAxRXIFG
63     * ~UCDORM -- Not dormant. All received characters will set UCAxRXIFG
64     * ~UCTXADDR -- Next frame transmitted is data
65     * ~UCTXBRK -- Next frame transmitted is not a break
66     * UCSWRST -- Enabled. USCI logic held in reset state
67     *
68     * Note: ~<BIT> indicates that <BIT> has value zero
69     */
70     UCA0CTL1 = UCSSEL_2 | UCSWRST;
71     /*
72     * Modulation Control Register
73     *
74     * UCBRF_0 -- First stage 0
75     * UCBRS_1 -- Second stage 1
76     * ~UCOS16 -- Disabled
77     *
78     * Note: ~UCOS16 indicates that UCOS16 has value zero
79     */
80     UCA0MCTL = UCBRF_0 | UCBRS_1;
```

```
80  /* Baud rate control register 0 */
81  UCA0BR0 = 104;
82  /* Enable USCI */
83  UCA0CTL1 &= ~UCSWRST;
84  /* USER CODE START (section: USCI_A0_graceInit_epilogue) */
85  /* User code */
86  /* USER CODE END (section: USCI_A0_graceInit_epilogue) */
87 }
```

sources/EE1054-Atividade05-USCI_A0_init.c

6 Microcontrolador Kinetis Freescale FRDM-KL25Z

6.1 Atividades propostas

Relatório de Experimentos – Microcontrolador família Kinetis Freescale

Atividade 6

1. Desenvolver um programa para realizar as seguintes funções com a placa FRDM KL25 fornecida:
 - a) Ao pressionar uma tecla acender o LED, ao pressionar a segunda vez o LED desliga.
 - b) Utilizar um timer para acender um LED, esperar um segundo com ele aceso e depois trocar para a segunda cor do LED, esperar mais um segundo e trocar novamente a cor, para a terceira cor do LED. Reiniciar o processo.
 - c) Utilizando o teclado touch da placa, fazer acender o LED e trocar de cores conforme o deslocamento pelo touch.

6.2 Relatório da Atividade

Nessa atividade foi escolhido elaborar os códigos a partir do compilador Mbed. Para isso é preciso gravar o bootloader adequado na FRDM KL25, ou seja, colocar a FRDM KL25 em modo de gravação de bootloader segurando o botão entre seus conectores, plugar a placa a um PC através do conector SDA e passar o arquivo. Após isso os códigos podem ser elaborados diretamente no site da Mbed que possui um compilador web que gera os arquivos a serem gravados na FRDM KL25 através da conexão USB como se estivesse passando arquivos para um Pendrive.

O passo a passo da gravação do bootloader da Mbed pode ser encontrado em seu site próprio¹.

Para fonte de consulta, inserimos o Diagrama de Bloco, como pode ser observado na Figura 28, extraída do Data Sheet do Kinetis Freescale. ([SEMICONDUTOR, 2013](#)).

¹ <<https://developer.mbed.org/handbook/mbed-FRDM-KL25Z-Getting-Started>>

Figura 28: Diagrama de Bloco do Data Sheet do Kinetis Freescale.



4 FRDM-KL25Z Hardware Overview

The features of the FRDM-KL25Z include:

- MKL25Z128VLK4 in an 80 LQFP package
- Capacitive touch slider
- MMA8451Q accelerometer
- Tri-color (RGB) LED
- Flexible power supply options – USB, coin cell battery, external source
- Battery-ready, power-measurement access points
- Easy access to MCU I/O via Arduino™ R3 compatible I/O connectors
- Programmable OpenSDA debug interface with multiple applications available including:
 - Mass storage device flash programming interface
 - P&E Debug interface provides run-control debugging and compatibility with IDE tools
 - CMSIS-DAP interface: new ARM standard for embedded debug interface
 - Data logging application

Figure 1 shows a block diagram of the FRDM-KL25Z design. The primary components and their placement on the hardware assembly are pointed out in Figure 2.

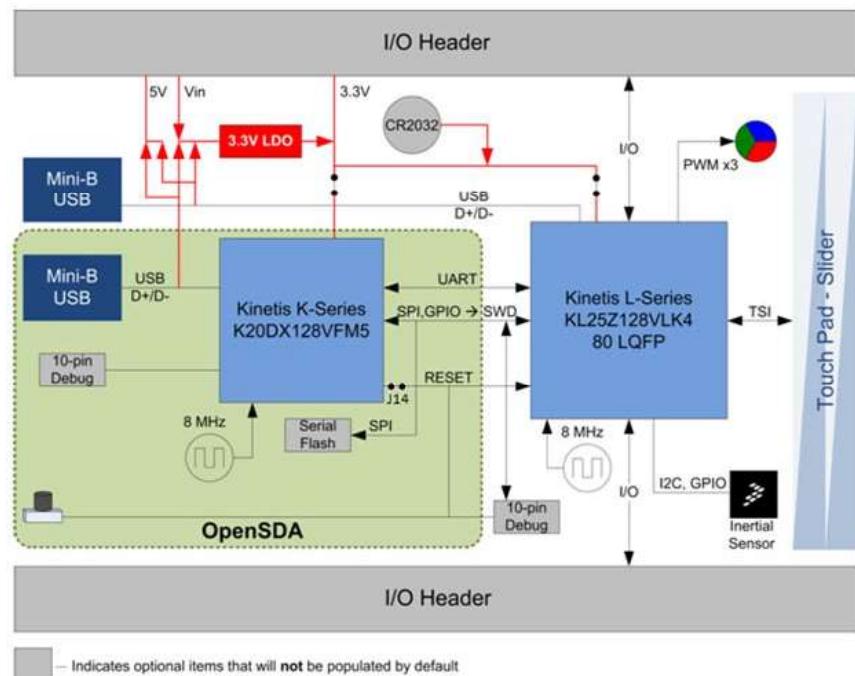


Figure 1. FRDM-KL25Z Block Diagram

Figura 26: Plataforma de desenvolvimento Kinetis Freescale utilizada na Atividade 6.

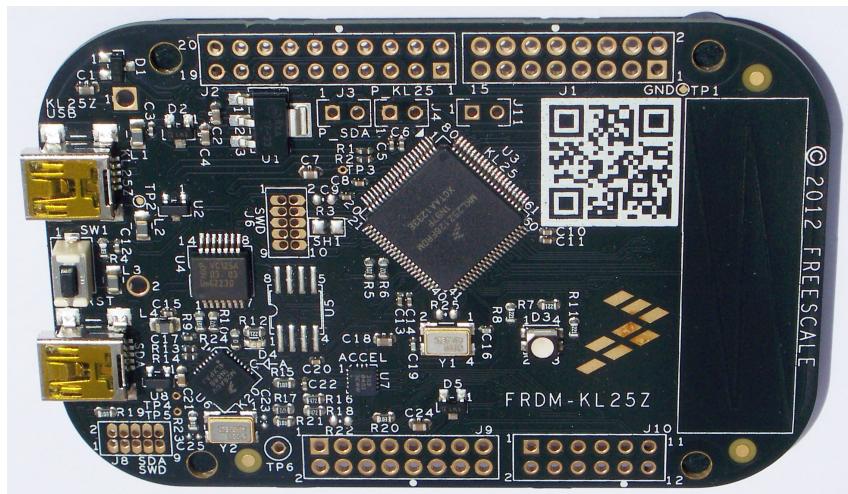
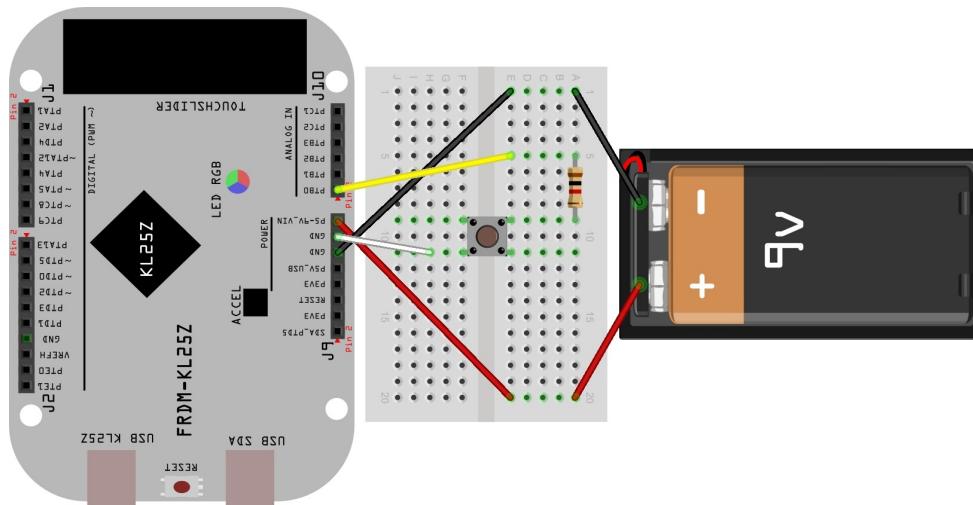


Figura 27: Diagrama Esquemático desenvolvido para o Kinetis Freescale durante a Atividade 6.

Fonte: Produzido pelos autores.



Os códigos-fontes elaborados estão disponibilizados a seguir:

Microcontrolador Kinetis Freescale – A06 (Códigos-fontes)

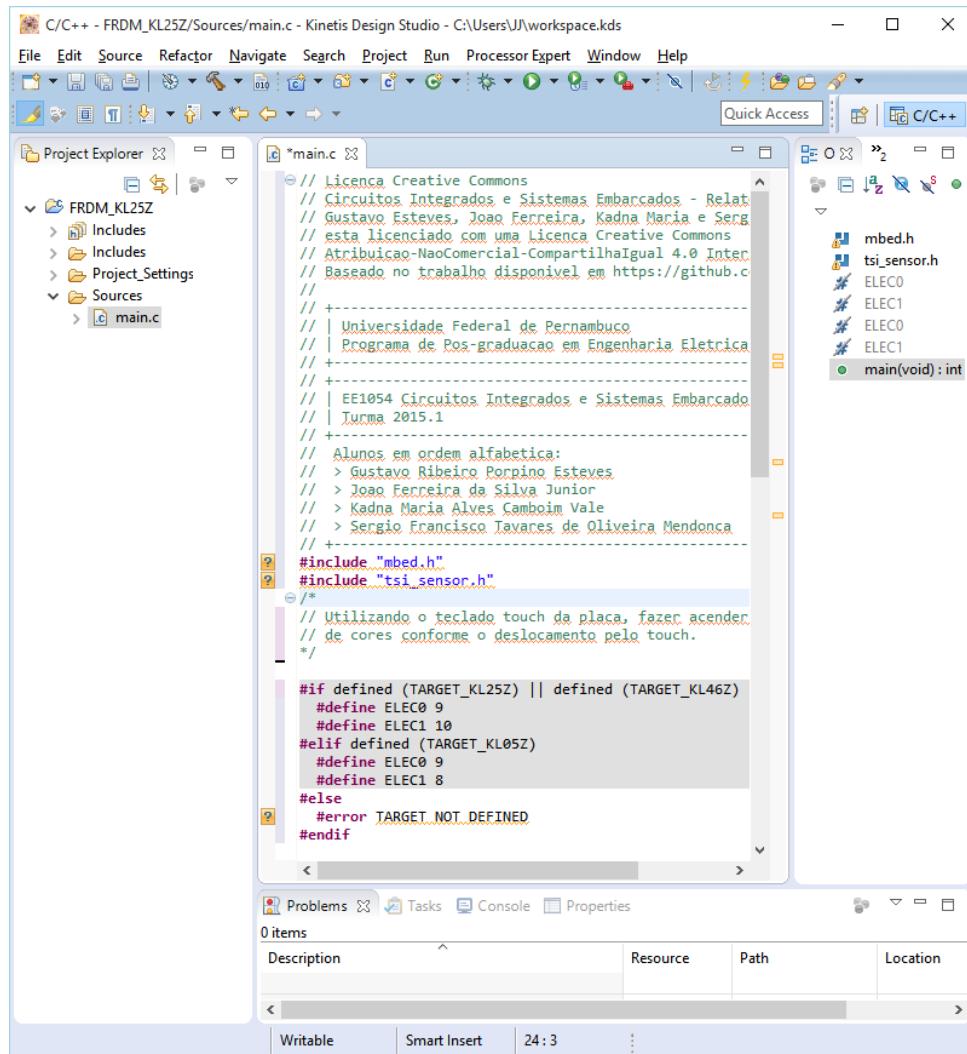
A06Q01-a

```

1 // Licenca Creative Commons
2 // Circuitos Integrados e Sistemas Embarcados - Relatorio Final de
3 // Gustavo Esteves, Joao Ferreira, Kadna Maria e Sergio Mendonca
4 // esta licenciado com uma Licenca Creative Commons
5 // Atribuicao-NaoComercial-CompartilhaIgual 4.0 Internacional.
6 // Baseado no trabalho disponivel em https://github.com/sftom/ee1054

```

Figura 29: Programação da placa Kinetis Freescale FRDM-KL25Z.



```

7 //+
8 // +-----+
9 // | Universidade Federal de Pernambuco          |
10 // | Programa de Pos-graduacao em Engenharia Eletrica |
11 // +-----+
12 // +-----+
13 // | EE1054 Circuitos Integrados e Sistemas Embarcados |
14 // | Turma 2015.1                                         |
15 // +-----+
16 // Alunos em ordem alfabetica:
17 // > Gustavo Ribeiro Porpino Esteves
18 // > Joao Ferreira da Silva Junior
19 // > Kadna Maria Alves Camboim Vale
20 // > Sergio Francisco Tavares de Oliveira Mendonca
21 // +-----+
22 #include "mbed.h"
23 /*
24 // Ao pressionar uma tecla acender o LED, ao pressionar a mesma tecla

```

```

25 // uma segunda vez apagar o LED.
26 */
27
28 #if defined (TARGET_KL25Z) || defined (TARGET_KL46Z)
29     #define ELECO 9
30     #define ELEC1 10
31 #elif defined (TARGET_KL05Z)
32     #define ELECO 9
33     #define ELEC1 8
34 #else
35     #error TARGET NOT DEFINED
36 #endif
37
38 int main(void) {
39     PE_low_level_init();
40
41     RED_SetVal(RED_DeviceData);
42     RED_ClrVal(RED_DeviceData);
43
44     for(;;) {
45         if (SW1_GetVal(NULL) == 0) { /* botao e pressionado */
46             RED_ClrVal(RED_DeviceData); /* LED acesso */
47             WAIT1_Waitms(1000);
48         }
49         if (SW1_GetVal(NULL) == 0) { /* botao e pressionado */
50             RED_SetVal(RED_DeviceData); /* LED apagado */
51             WAIT1_Waitms(1000);
52         }
53     }
54 }
```

sources/EE1054-Atividade06-Questao01a-Led.cpp

A06Q01-b

```

1 // Licenca Creative Commons
2 // Circuitos Integrados e Sistemas Embarcados - Relatorio Final de
3 // Gustavo Esteves, Joao Ferreira, Kadna Maria e Sergio Mendonca
4 // esta licenciado com uma Licenca Creative Commons
5 // Atribuicao-NaoComercial-CompartilhaIgual 4.0 Internacional.
6 // Baseado no trabalho disponivel em https://github.com/sftom/ee1054
7 //
8 // +-----+
9 // | Universidade Federal de Pernambuco | 
10 // | Programa de Pos-graduacao em Engenharia Eletrica | 
11 // +-----+
12 // +-----+
```

```

13 // | EE1054 Circuitos Integrados e Sistemas Embarcados |
14 // | Turma 2015.1 |
15 // +-----+
16 // Alunos em ordem alfabetica:
17 // > Gustavo Ribeiro Porpino Esteves
18 // > Joao Ferreira da Silva Junior
19 // > Kadna Maria Alves Camboim Vale
20 // > Sergio Francisco Tavares de Oliveira Mendonca
21 // +-----+
22 #include "mbed.h"
23 /*
24 // Utilizando um timer para acender um LED, esperar um segundo com ele
25 // acesso e depois trocar para a segunda cor do LED, esperar mais um
26 // segundo e trocar novamente a cor, para a terceira cor do LED.
27 // Reiniciar o processo.
28 */
29 DigitalOut L1(LED1);
30 DigitalOut L2(LED2);
31 DigitalOut L3(LED3);
32
33 int main() {
34     L1 = 1;
35     L2 = 1;
36     L3 = 1;
37     while(1) {
38         L1 = 0;
39         wait(1);
40         L1 = 1;
41         wait(1);
42         L2 = 0;
43         wait(1);
44         L2 = 1;
45         wait(1);
46         L3 = 0;
47         wait(1);
48         L3 = 1;
49         wait(1);
50     }
51 }
```

sources/EE1054-Atividade06-Questao01b-Timer.cpp

A06Q01-c

```

1 // Licenca Creative Commons
2 // Circuitos Integrados e Sistemas Embarcados - Relatorio Final de
3 // Gustavo Esteves, Joao Ferreira, Kadna Maria e Sergio Mendonca
```

```
4 // esta licenciado com uma Licenca Creative Commons
5 // Atribuicao-NaoComercial-CompartilhaIgual 4.0 Internacional.
6 // Baseado no trabalho disponivel em https://github.com/sftom/ee1054
7 //
8 // +-----+
9 // | Universidade Federal de Pernambuco           |
10 // | Programa de Pos-graduacao em Engenharia Eletrica |
11 // +-----+
12 // +-----+
13 // | EE1054 Circuitos Integrados e Sistemas Embarcados |
14 // | Turma 2015.1                                         |
15 // +-----+
16 // Alunos em ordem alfabetica:
17 // > Gustavo Ribeiro Porpino Esteves
18 // > Joao Ferreira da Silva Junior
19 // > Kadna Maria Alves Camboim Vale
20 // > Sergio Francisco Tavares de Oliveira Mendonca
21 // +-----+
22 #include "mbed.h"
23 #include "tsi_sensor.h"
24 /*
25 // Utilizando o teclado touch da placa, fazer acender o led e trocar
26 // de cores conforme o deslocamento pelo touch.
27 */
28
29 #if defined (TARGET_KL25Z) || defined (TARGET_KL46Z)
30     #define ELEC0 9
31     #define ELEC1 10
32 #elif defined (TARGET_KL05Z)
33     #define ELEC0 9
34     #define ELEC1 8
35 #else
36     #error TARGET NOT DEFINED
37 #endif
38
39 int main(void) {
40     PwmOut R(LED_RED);
41     PwmOut G(LED_GREEN);
42     PwmOut B(LED_BLUE);
43     TSIAnalogSlider tsi(ELEC0, ELEC1, 1);
44     R = 1;
45     G = 1;
46     B = 1;
47     while (true) {
48         int x = tsi.readPercentage()*100;
49         if (x > 1 && x < 20) {
50             R = 0;
```

```
51     G = 1;
52     B = 1;
53     wait(0.1);
54 }
55 if (x > 20 && x < 40) {
56     R = 1;
57     G = 0;
58     B = 1;
59     wait(0.1);
60 }
61 if (x > 40 && x < 60) {
62     R = 1;
63     G = 1;
64     B = 0;
65     wait(0.1);
66 }
67 if (x > 60 && x < 80) {
68     R = 0;
69     G = 0;
70     B = 1;
71     wait(0.1);
72 }
73 if (x > 80) {
74     R = 0;
75     G = 1;
76     B = 0;
77     wait(0.1);
78 }
79 if (x < 1) {
80     R = 1;
81     G = 1;
82     B = 1;
83     wait(0.1);
84 }
85 }
86 }
```

sources/EE1054–Atividade06–Questao01c–Touch.cpp

7 Placa Raspberry Pi B+

7.1 Atividades propostas

Trabalho final da disciplina

Atividade 7

1. Utilizando a placa Raspberry Pi, instalar o sistema operacional, linguagem Python 3 e editor de código.
2. Fazer um programa em Python para apresentar na tela a informação da disciplina e dos integrantes do grupo.
3. Fazer um programa para ler a temperatura e umidade, através do sensor fornecido. Apresentar os resultados em tela.
4. Fazer um programa para configurar o display de cristal líquido fornecido. Apresentar a mesma informação de temperatura e umidade na tela do display.
5. Apresentar um relatório completo sobre todos os itens solicitados com o esquemático de ligação dos componentes.

OBS: Para finalização da disciplina pede-se:

1. Relatório de todos os trabalhos desenvolvidos, informando a tecnologia utilizada, compilador, etc (PDF e DOC).
2. Anexo com todos os códigos desenvolvidos nos trabalhos (todos os códigos fontes).
3. Gravação do material em um DVD, identificando os componentes do grupo.

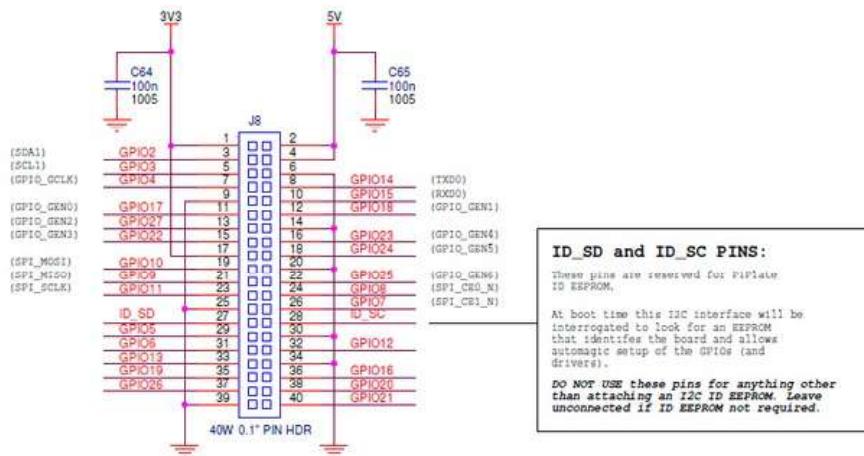
7.2 Relatório da Atividade

Primeiramente o sistema operacional Raspbian foi gravado em um cartão SD seguindo os passos que podem ser encontrados no Manual¹.

Para fonte de consulta, inserimos o Diagrama Pinout da Placa Raspberry Pi B+, como pode ser observado na Figura 33, extraída do Manual do Usuário. ([ADAFRUIT, 2015](#)).

¹ <<https://www.raspberrypi.org/help/noobs-setup>>

Figura 33: Diagrama Pinout da Placa Raspberry Pi B+



First thing to notice, **the top 26 pins of the 40-pin connector are the same as the original**. That means that most/many Pi Plates that plug into the Model B will plug into the B+ just fine. They won't sit in the same location - they'll be slid down just a bit but electrically-wise it's the same.

New GPIOs

Figura 30: Raspberry B+ utilizada na Atividade 7.

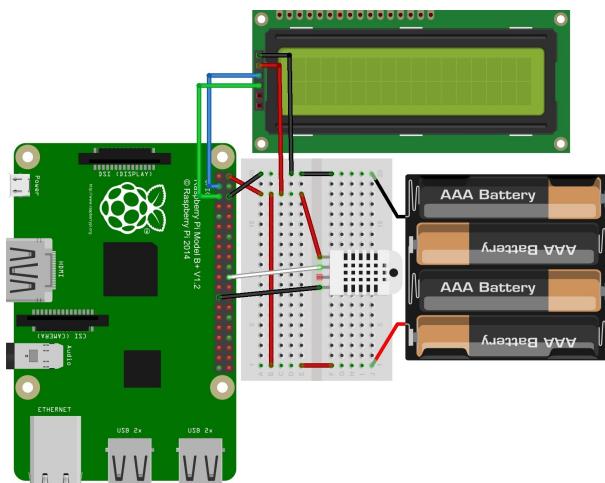


Figura 31: Modelo de LCD utilizado na prática com Raspberry Pi B+.



Figura 32: Diagrama Esquemático desenvolvido para o Raspberry Pi B+ durante a Atividade 7.

Fonte: Produzido pelos autores.



Os códigos-fontes elaborados estão disponibilizados a seguir:

Figura 34: Programação da placa Raspberry Pi B+ no ambiente de programação IDLE do Python 3.

```

76 EE1054-Atividade07-Questao03-DHT.py
File Edit Format Run Options Windows Help
#!/bin/python
# -*- coding: cp1252 -*-
...
// Licenca Creative Commons
// Circuitos Integrados e Sistemas Embarcados - Relatorio Final de
// Gustavo Esteves, Joao Ferreira, Kadna Maria e Sergio Mendonca
// esta licenciado com uma Licenca Creative Commons
// Atribuicao-NaoComercial-CompartilhaIgual 4.0 Internacional.
// Baseado no trabalho disponivel em https://github.com/sftom/ee1054
//
// +-----+
// | Universidade Federal de Pernambuco |
// | Programa de Pos-graduacao em Engenharia Eletrica |
// +-----+
// +-----+
// | EE1054 Circuitos Integrados e Sistemas Embarcados |
// | Turma 2015.1 |
// +-----+
// Alunos em ordem alfabetica:
// > Gustavo Ribeiro Porpino Esteves
// > Joao Ferreira da Silva Junior
// > Kadna Maria Alves Camboim Vale
// > Sergio Francisco Tavares de Oliveira Mendonca
// +-----+
...
import time
import sys
import subprocess
import platform
# Bloco de bibliotecas do usuario
import Adafruit_DHT
import RPi.GPIO as GPIO
...
Fazer um programa para ler a temperatura e umidade, atraves do sensor
fornecido. Apresentar os resultados em tela.
...

# Define o tipo de sensor DHT11
sensor = Adafruit_DHT.DHT11
# Define a GPIO conectada ao pino de dados do sensor

```

Ln: 1 Col: 0

Placa Raspberry Pi – A07 (Códigos-fontes)

A07Q02

```

1 #!/bin/python
2 # -*- coding: cp1252 -*-
3 ''
4 // Licenca Creative Commons
5 // Circuitos Integrados e Sistemas Embarcados - Relatorio Final de
6 // Gustavo Esteves, Joao Ferreira, Kadna Maria e Sergio Mendonca
7 // esta licenciado com uma Licenca Creative Commons
8 // Atribuicao-NaoComercial-CompartilhaIgual 4.0 Internacional.
9 // Baseado no trabalho disponivel em https://github.com/sftom/ee1054
10 //
11 // +-----+
12 // | Universidade Federal de Pernambuco |
13 // | Programa de Pos-graduacao em Engenharia Eletrica |
14 // +-----+

```

```
15 // +-----+
16 // | EE1054 Circuitos Integrados e Sistemas Embarcados |
17 // | Turma 2015.1 |
18 // +-----+
19 // Alunos em ordem alfabetica:
20 // > Gustavo Ribeiro Porpino Esteves
21 // > Joao Ferreira da Silva Junior
22 // > Kadna Maria Alves Camboim Vale
23 // > Sergio Francisco Tavares de Oliveira Mendonca
24 // +-----+
25 """
26 import time
27 import sys
28 import subprocess
29 import platform
30 """
31 Fazer um programa em Python para apresentar na tela a informacao
32 da disciplina e dos integrantes do grupo.
33 """
34
35 """
36 Imprime o cabecalho de informacoes
37 """
38 def cabecalho():
39     # Limpa o terminal
40     subprocess.Popen(
41         "cls" if platform.system() == "Windows"
42         else "clear", shell=True
43     )
44     print ("+-----+")
45     print ("| Universidade Federal de Pernambuco | ")
46     print ("| Programa de Pos-graduacao em Engenharia Eletrica | ")
47     print ("+-----+")
48     print ("+-----+")
49     print ("| EE1054 Circuitos Integrados e Sistemas Embarcados | ")
50     print ("| Turma 2015.1 | ")
51     print ("+-----+")
52     print (" Alunos em ordem alfabetica:")
53     time.sleep(1)
54     print (" > Gustavo Ribeiro Porpino Esteves")
55     time.sleep(1)
56     print (" > Joao Ferreira da Silva Junior")
57     time.sleep(1)
58     print (" > Kadna Maria Alves Camboim Vale")
59     time.sleep(1)
60     print (" > Sergio Francisco Tavares de Oliveira Mendonca")
61     print ("+-----+")
```

```
62
63 # Bloco de repeticao para chamada do cabecalho
64 while(1):
65     # Chama a funcao para imprimir o cabecalho
66     cabecalho()
```

sources/EE1054-Atividade07-Questao02-Print.py

A07Q03

```
1#!/bin/python
2# -*- coding: cp1252 -*-
3"""
4// Licenca Creative Commons
5// Circuitos Integrados e Sistemas Embarcados - Relatorio Final de
6// Gustavo Esteves, Joao Ferreira, Kadna Maria e Sergio Mendonca
7// esta licenciado com uma Licenca Creative Commons
8// Atribuicao-NaoComercial-CompartilhaIgual 4.0 Internacional.
9// Baseado no trabalho disponivel em https://github.com/sftom/ee1054
10//
11// +-----+
12// | Universidade Federal de Pernambuco           |
13// | Programa de Pos-graduacao em Engenharia Eletrica |
14// +-----+
15// +-----+
16// | EE1054 Circuitos Integrados e Sistemas Embarcados |
17// | Turma 2015.1                                         |
18// +-----+
19// Alunos em ordem alfabetica:
20// > Gustavo Ribeiro Porpino Esteves
21// > Joao Ferreira da Silva Junior
22// > Kadna Maria Alves Camboim Vale
23// > Sergio Francisco Tavares de Oliveira Mendonca
24// +-----+
25"""
26import time
27import sys
28import subprocess
29import platform
30# Bloco de bibliotecas do usuario
31import Adafruit_DHT
32import RPi.GPIO as GPIO
33"""
34Fazer um programa para ler a temperatura e umidade, atraves do sensor
35fornecido. Apresentar os resultados em tela.
36"""
37
```

```
38 # Define o tipo de sensor DHT11
39 sensor = Adafruit_DHT.DHT11
40 # Define a GPIO conectada ao pino de dados do sensor
41 GPIO.setmode(GPIO.BOARD)
42 pino_sensor = 25
43
44 '''
45 Imprime o cabecalho de informacoes
46 '''
47 def cabecalho():
48     # Limpa o terminal
49     subprocess.Popen(
50         "cls" if platform.system() == "Windows"
51         else "clear", shell=True
52     )
53     print ("+-----+")
54     print ("| Universidade Federal de Pernambuco | ")
55     print ("| Programa de Pos-graduacao em Engenharia Eletrica | ")
56     print ("+-----+")
57     print ("+-----+")
58     print ("| EE1054 Circuitos Integrados e Sistemas Embarcados | ")
59     print ("| Turma 2015.1 | ")
60     print ("+-----+")
61     print (" Alunos em ordem alfabetica:")
62     time.sleep(1)
63     print (" > Gustavo Ribeiro Porpino Esteves")
64     time.sleep(1)
65     print (" > Joao Ferreira da Silva Junior")
66     time.sleep(1)
67     print (" > Kadna Maria Alves Camboim Vale")
68     time.sleep(1)
69     print (" > Sergio Francisco Tavares de Oliveira Mendonca")
70     print ("+-----+")
71
72 # Bloco de comandos para leitura do sensor
73 while(1):
74     # Efetua a leitura do sensor
75     umid, temp = Adafruit_DHT.read_retry(sensor, pino_sensor);
76     # Chama a funcao para imprimir o cabecalho
77     cabecalho()
78     # Caso leitura esteja ok, mostra os valores na tela
79     if umid is not None and temp is not None:
80         print ("+-----+")
81         print ("| Temperatura = {0:0.1f}").format(temp)
82         print ("| Umidade      = {1:0.1f}").format(umid)
83         print ("+-----+")
84         time.sleep(2)
```

```

85 else:
86     #Mensagem de erro de comunicacao com o sensor
87     print ("+-----+")
88     print ("| Falha ao ler dados do DHT11           |")
89     print ("+-----+")
90     time.sleep(2)

```

sources/EE1054-Atividade07-Questao03-DHT.py

A07Q04

```

1#!/bin/python
2# -*- coding: cp1252 -*-
3'''
4// Licenca Creative Commons
5// Circuitos Integrados e Sistemas Embarcados - Relatorio Final de
6// Gustavo Esteves, Joao Ferreira, Kadna Maria e Sergio Mendonca
7// esta licenciado com uma Licenca Creative Commons
8// Atribuicao-NaoComercial-CompartilhaIgual 4.0 Internacional.
9// Baseado no trabalho disponivel em https://github.com/sftom/ee1054
10//
11// +-----+
12// | Universidade Federal de Pernambuco          |
13// | Programa de Pos-graduacao em Engenharia Eletrica |
14// +-----+
15// +-----+
16// | EE1054 Circuitos Integrados e Sistemas Embarcados |
17// | Turma 2015.1                                     |
18// +-----+
19// Alunos em ordem alfabetica:
20// > Gustavo Ribeiro Porpino Esteves
21// > Joao Ferreira da Silva Junior
22// > Kadna Maria Alves Camboim Vale
23// > Sergio Francisco Tavares de Oliveira Mendonca
24// +-----+
25'''
26import time
27import sys
28import subprocess
29import platform
30import serial
31# Bloco de bibliotecas do usuario
32import Adafruit_DHT
33import RPi.GPIO as GPIO
34'''
35Fazer um programa para configurar o display de cristal liquido
36fornecido. Apresentar a mesma informacao de temperatura e umidade na

```

```
37 tela do display.
38 """
39
40 # Define o tipo de sensor DHT11
41 sensor = Adafruit_DHT.DHT11
42 # Define a GPIO conectada ao pino de dados do sensor
43 GPIO.setmode(GPIO.BOARD)
44 pino_sensor = 25
45
46 # Definicoes para comunicacao serial com o display
47 port = serial.Serial("/dev/ttyAMA0", baudrate=9600, timeout=1)
48
49 """
50 Imprime o cabecalho de informacoes
51 """
52 def cabecalho():
53     # Limpa o terminal
54     subprocess.Popen(
55         "cls" if platform.system() == "Windows"
56         else "clear", shell=True
57     )
58     print ("+-----+")
59     print ("| Universidade Federal de Pernambuco | ")
60     print ("| Programa de Pos-graduacao em Engenharia Eletrica | ")
61     print ("+-----+")
62     print ("+-----+")
63     print ("| EE1054 Circuitos Integrados e Sistemas Embarcados | ")
64     print ("| Turma 2015.1 | ")
65     print ("+-----+")
66     print (" Alunos em ordem alfabetica:")
67     time.sleep(1)
68     print (" > Gustavo Ribeiro Porpino Esteves")
69     time.sleep(1)
70     print (" > Joao Ferreira da Silva Junior")
71     time.sleep(1)
72     print (" > Kadna Maria Alves Camboim Vale")
73     time.sleep(1)
74     print (" > Sergio Francisco Tavares de Oliveira Mendonca")
75     print ("+-----+")
76
77 # Bloco de comandos para leitura do sensor
78 while(1):
79     # Efetua a leitura do sensor
80     umid, temp = Adafruit_DHT.read_retry(sensor, pino_sensor);
81     # Chama a funcao para imprimir o cabecalho
82     cabecalho()
83     # Inicializa
```

```
84 port.write('\x0d')
85 # Limpa o LCD
86 port.write('\xfe\x01')
87 time.sleep(0.1)
88 port.write("Temperatura")
89 port.write("{0:0.1f}").format(temp)
90 time.sleep(0.5)
91 port.write("Umidade")
92 port.write("{1:0.1f}").format(umid)
93 time.sleep(0.5)
```

sources/EE1054-Atividade07-Questao04-Display.py

8 Placa Intel Galileo 2nd Gen.

8.1 Atividades propostas

Relatório de experimentos – Aula prática com a placa Intel Galileo

Atividade 8

1. Instalar a ferramenta (IDE) para trabalhar com a placa Galileo e Android.
2. Realizar testes para piscar LEDs com a placa Galileo, utilizando o LED da própria placa.
3. Fazer um programa para gerar pulsos em um pino de saída, bem como acender um indicador (LED) ao mesmo tempo, utilizando o Shield Arduíno. Utilizar duas teclas externas uma para aumentar a velocidade das piscadas do LED e a outra para diminuir a velocidade das piscadas do LED. Pede-se o controle de velocidade de 1 a 60 piscadas.

8.2 Relatório da Atividade

Para fonte de consulta, inserimos o Diagrama de conexões da Placa Intel Galileo, como pode ser observado na Figura 37, extraída do Guia do Usuário da Placa Intel Galileo 2nd Gen. ([CORP., 2014](#)).

Figura 35: Atividade prática na Plataforma de Desenvolvimento Intel Galileo



Figura 37: Diagrama de conexões da Placa Intel Galileo.

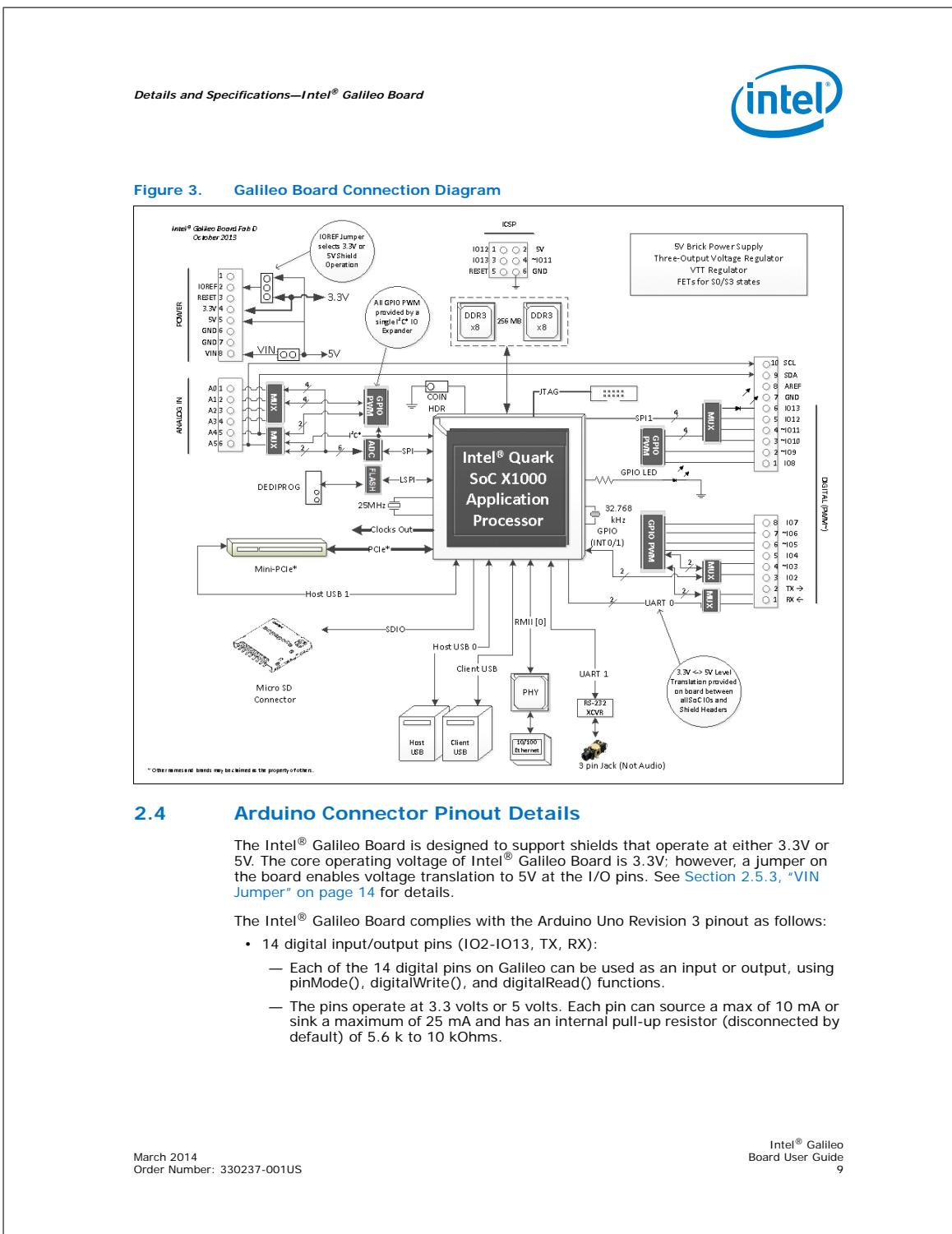
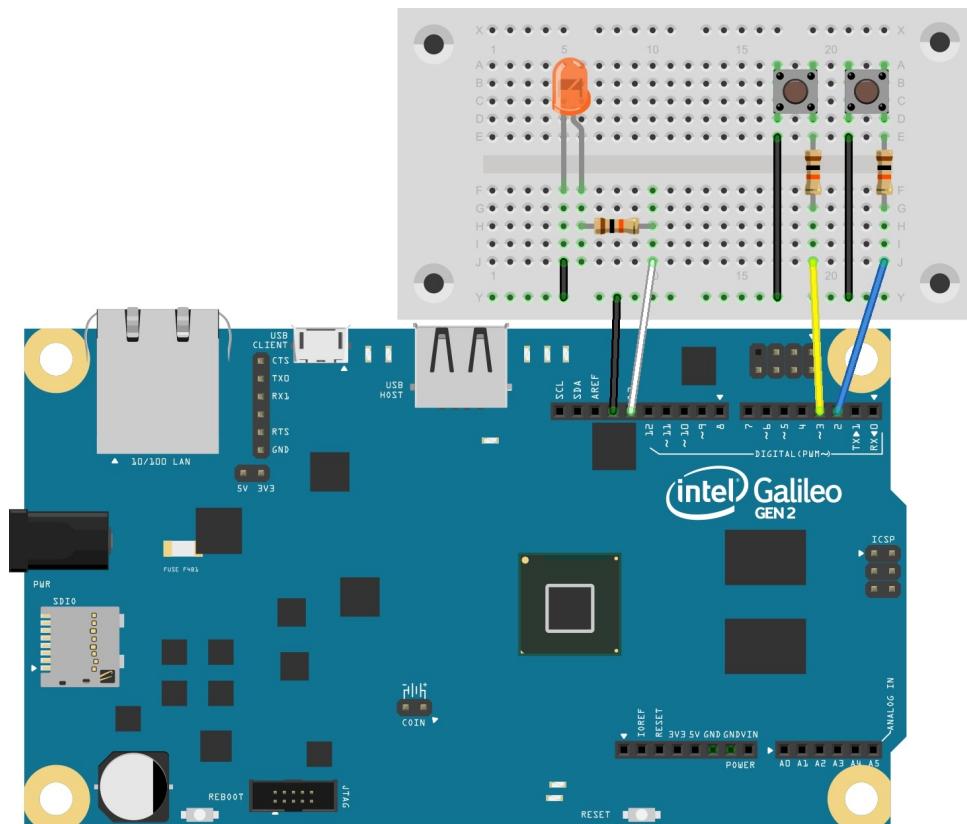


Figura 36: Diagrama Esquemático desenvolvido para o Intel Galileo durante a Atividade 8.

Fonte: Produzido pelos autores.



Os códigos-fontes elaborados estão disponibilizados a seguir:

Placa Intel Galileo – A08 (Códigos-fontes)

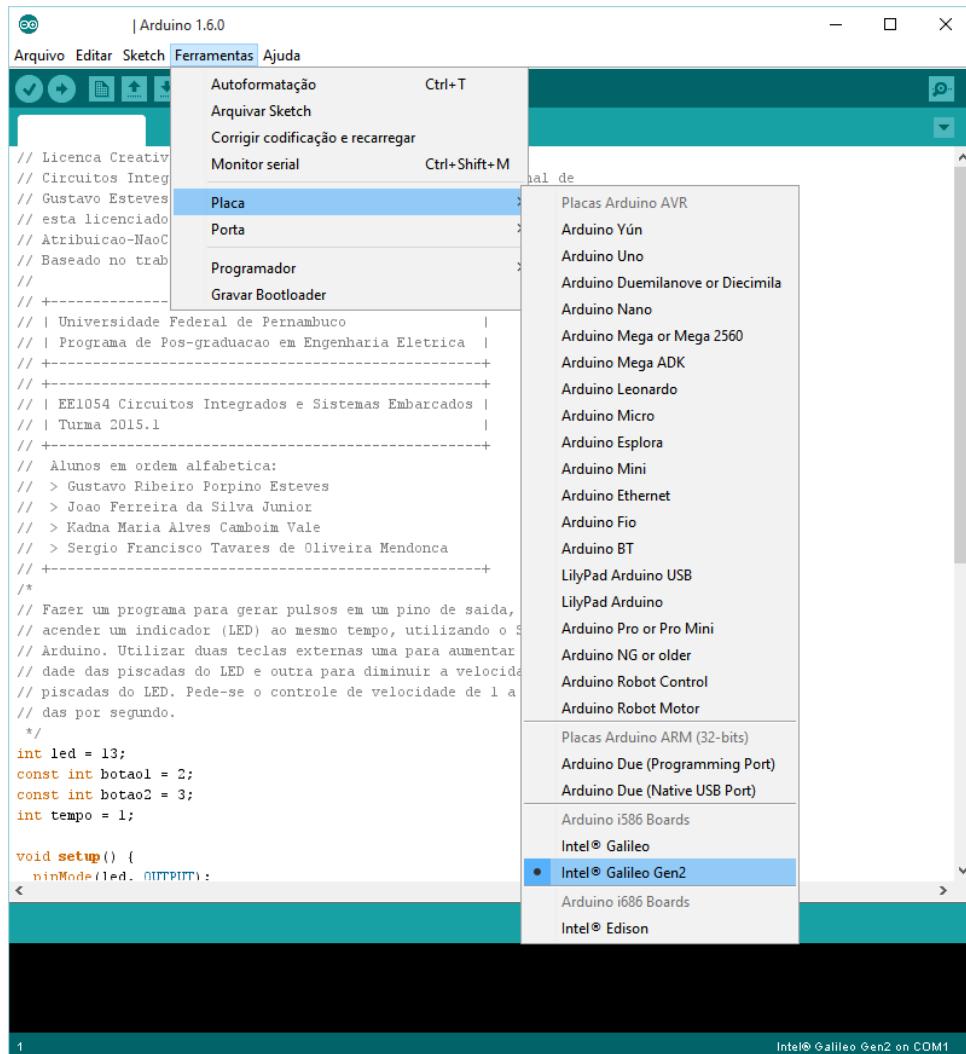
A08Q02-03

```

1 // Licenca Creative Commons
2 // Circuitos Integrados e Sistemas Embarcados - Relatorio Final de
3 // Gustavo Esteves, Joao Ferreira, Kadna Maria e Sergio Mendonca
4 // esta licenciado com uma Licenca Creative Commons
5 // Atribuicao-NaoComercial-CompartilhaIgual 4.0 Internacional.
6 // Baseado no trabalho disponivel em https://github.com/sftom/ee1054
7 //
8 // +-----+
9 // | Universidade Federal de Pernambuco | 
10 // | Programa de Pos-graduacao em Engenharia Eletrica | 
11 // +-----+
12 // +-----+
13 // | EE1054 Circuitos Integrados e Sistemas Embarcados | 
14 // | Turma 2015.1 | 

```

Figura 38: Programação da placa Intel Galileo 2nd Gen. através da IDE do Arduino.



```

15 // +-----+
16 // Alunos em ordem alfabetica:
17 // > Gustavo Ribeiro Porpino Esteves
18 // > Joao Ferreira da Silva Junior
19 // > Kadna Maria Alves Camboim Vale
20 // > Sergio Francisco Tavares de Oliveira Mendonca
21 // +-----+
22 */
23 // Fazer um programa para gerar pulsos em um pino de saida, bem como
24 // acender um indicador (LED) ao mesmo tempo, utilizando o Shield
25 // Arduino. Utilizar duas teclas externas uma para aumentar a veloci-
26 // dade das piscadas do LED e outra para diminuir a velocidade das
27 // piscadas do LED. Pede-se o controle de velocidade de 1 a 60 pisca-
28 // das por segundo.
29 */
30 int led = 13;
31 const int botao1 = 2;
32 const int botao2 = 3;

```

```
33 int tempo = 1;
34
35 void setup() {
36     pinMode(led, OUTPUT);
37     pinMode(botao1, INPUT);
38     pinMode(botao2, INPUT);
39 }
40
41 void loop() {
42     if (digitalRead(botao1) == HIGH) {
43         tempo++;
44         while(digitalRead(botao1) == HIGH){;}
45     }
46     if (digitalRead(botao2) == HIGH) {
47         tempo--;
48         while(digitalRead(botao2) == HIGH){;}
49     }
50     if (tempo > 60) {
51         tempo = 60;
52     }
53     if (tempo < 1) {
54         tempo = 1;
55     }
56     digitalWrite(led, HIGH);
57     delay(tempo * 17);
58     digitalWrite(led, LOW);
59     delay(tempo * 17);
60
61 }
```

sources/EE1054-Atividade08-BlinkGalileo.ino

Referências

ADAFRUIT. *Introducing the Raspberry Pi Model B+*. 2015. 34 p. Disponível em: <<https://learn.adafruit.com/downloads/pdf/introducing-the-raspberry-pi-model-b-plus-plus-differences-vs-model-b.pdf>>. Acesso em: 30 jun. 2015. Citado na página 72.

CORP., A. *8-bit Microcontroller with 4K Bytes Flash - AT89C51*. 2000. 17 p. Disponível em: <<http://www.atmel.com/images/doc0265.pdf>>. Acesso em: 15 jun. 2015. Citado 2 vezes nas páginas 14 e 28.

CORP., I. *Intel Galileo Board User Guide*. 2014. 19 p. Disponível em: <<http://www.intel.com/content/dam/www/public/us/en/documents/guides/galileo-user-guide.pdf>>. Acesso em: 20 jul. 2015. Citado na página 82.

INSTRUMENTS, T. *Mixed Signal Microcontroller Datasheet MSP430F22x2 and MSP430F22x4*. 2012. 94 p. Disponível em: <<http://www.ti.com/lit/ds/symlink/msp430f2274.pdf>>. Acesso em: 21 jun. 2015. Citado 2 vezes nas páginas 45 e 51.

SEMICONDUTOR, F. *Freescale FRDM-KL25z User's Manual*. 2013. 20 p. Disponível em: <<http://www.freescale.com/FRDM-KL25Z>>. Acesso em: 9 jun. 2015. Citado na página 64.

TECHNOLOGY, M. *PIC18F2455/2550/4455/4550 Data Sheet*. 2009. 438 p. Disponível em: <<http://ww1.microchip.com/downloads/en/DeviceDoc/39632D.pdf>>. Acesso em: 6 jul. 2015. Citado na página 35.