ESZI041-18-Prog. de Software Embarcado

CECS / Engenharia de Informação

Prof. João Ranhel **– Prova / trabalho do LABORATÓRIO:**

Desenvolva um projeto levando em conta as práticas que fizemos, associada a detalhes do seu RA. O projeto será feito usando *FreeRTOS* – podem usar rotinas desenvolvidas em aulas.

Para *piscar* os LEDs use *dt* = 200 ms ON/OFF. Os kits se comunicam via ***UART***: ***RX*** = ***PA10***, ***TX*** = ***PA9*** (BAUD RATE = 115.200 / 8 bits / no parity/ 1 stopBit). Detalhes do projeto:

a.1) A partir do *RESET* da MCU, monte uma rotina de teste para ver se todos os LEDs do painel de display (4x7-SEG) estão funcionando no *shield*. O teste deve ligar todos os segmentos no display:  .

a.2) Depois de 2s com tudo aceso, mostre no display os 4 dígitos menos significativos do seu RA, com pontos decimais desligados. Criatividade conta! (*a biblioteca de acionamento dos displays será fornecida e explicada*).

b) Ao mesmo tempo, assim que você acionar o *RESET*, o programa deve disparar:

b.1) um ***cronômetro***, ***decrescente*** se o dígito menos significativo do seu RA for PAR, e ***crescente*** se for IMPAR;

b.2) disparar também um ***conversor ADC*** que amostra continuamente, vamos ler um valor analógico no pino ***PA0***. A taxa de amostragem do ADC (samples/s) será: *taxa* = 10+β*10; β=2º dígito do seu RA. Ex: RA termina em ..***2***6, o ADC fará *taxa*=10+***2***x10=30 samples/s. Se ..*4*7, *taxa*=10+***4***x10=50 s/s.

c) Ao acionarmos o botão ***A1*** (***PA1***) no *shield Arduino*, o display de 7-SEG alterna o valor do cronômetro e do ADC a cada 4 segundos (*2v/4s*: 2 *valores*/4 *segs* – ou: cronômetro por 4s, ADC por 4s, loop contínuo).

c.1) O cronômetro será mostrado no painel display na forma: ***minutos*** (1 dígito), "**pto** decimal" ligado, ***segundos*** (com 2 dígitos), "**pto** decimal" ligado, ***décimos*** de segundo (1 dígito):  ;

c.2) O ADC será mostrado na forma: ***unidade***, "**pto**" ligado, ***centésimo***, ***décimo***, e ***milésimo*** de Volt com pontos desligados (ou seja, o milésimo é o menos significativo, resolução = 0.00***1*** Volt):  ;

c.3) para distinguir o valor mostrado no display, *pisque* o ***LED 1*** (*PB15*) quando mostrar o cronômetro, e *pisque* o ***LED 2*** (*PB14*) quando mostrar o valor do ADC. Claro, o formato dos valores no display 7-SEG já indica qual medida está no visor, mas usaremos os LEDs piscando também;

d) Ao acionarmos o botão ***A2*** (*PA2*), ***seu kit deve iniciar comunicação com outra placa*** e deve, a partir daí, enviar dados para a placa de um colega. Você passa a '*requisitar serviço*' da outra placa.

d.1) Se você acionar A2 antes de acionar A1 seus valores serão mostrados no kit do colega, mas não no seu kit; ou seja, sua placa continuará a mostrar seu RA, e seus dados são mostrados apenas remotamente.

e) quando a placa de outro colega solicitar serviço (mostrar os dados dele), ***sua placa NÃO poderá recusar*** o pedido e deve mostrar dados no modo 4 valores/2 segs (*4v/2s*); mostrando, repetidamente, o seguinte:

* por 2 segundos, piscando o D1, deve mostrar seu próprio cronômetro (ou RA);
** por 2 segundos, piscando o D2, deve mostrar seu próprio ADC (ou RA);
*** por 2 segundos, piscando o ***LED D3*** - *PB13*, deve mostrar o valor do cronômetro do colega;
**** por 2 segundos, piscando o ***LED D4*** - *PB12*, deve mostrar o valor do ADC do colega;

e.1) Se você ainda não acionou A1 e sua placa recebe uma solicitação de *serviço* ela muda o modo do display para *4v/2s*, mostrando seu RA no lugar do seu cronômetro e ADC, e em seguida os dados do outro.

e.2) Será possível deixar de atender esse *serviço* (deixar de mostrar dados do outro e sair do modo *4v/2s*) acionando ***A1*** na sua placa. Isso muda o modo do display formato *2v/4s* (item '*c*' anterior). Seu programa deve enviar uma mensagem avisando que não está mais prestando o serviço. Contudo, se o outro kit enviar novamente a solicitação (ou seja, se pressionar ***A2*** na outra placa), sua placa tem que voltar a atender.

f) Claro que a placa do colega também NÃO pode rejeitar uma solicitação de comunicação se você acionar ***A2*** na sua placa. Portanto, ao acionar ***A2***, ***SUA PLACA deve começar a enviar dados*** para a placa do colega;

g) quando o ***A2*** for acionado em AMBAS as placas, a ***comunicação deve ocorrer de forma cruzada*** – ou seja, sua placa mostra os dados da placa do colega e a placa dele mostra seus dados, ambas no formato *4v/2s*, como descrito no item '*e*' acima.

h) ao acionarmos o botão ***A3*** (***PA3***) você deixa de solicitar o serviço da placa do colega (ou vice-versa, se o colega acionar ***A3*** na placa dele). Ao acionar ***A3***, você libera a placa do colega para voltar a mostrar só os dados dele (vice-versa). Em outras palavras, ao acionar ***A3*** você deve enviar uma mensagem para o outro kit informando que não quer mais o serviço (vice-versa).

h.1) assim, se o colega acionar ***A3*** na outra placa, você deixa de mostrar o valor dele e voltará para o modo de amostragem local. Se você ainda tiver acionado ***A1***, sua placa volta a mostrar seu RA; mas se já estava mostrando seus dados, então volta ao modo de amostragem *2v/4s* descrito no item '*c*'.

h.2) note que, o botão ***A3 dispensa sua requisição de serviço***; mas não libera você de atender o serviço da outra placa se estiver ativo (como é o caso quando acionamos ***A1***). Ou seja, se ***a outra*** acionar ***A3***, a sua placa ficará liberada de mostrar os dados dele; mas se você acionar ***A3***, e a outra placa ainda estiver requisitando o serviço, seu kit deve mostrar os dados dela, como descrito no item '*e*'.

i) A turma (toda a classe) deve decidir COMO será a comunicação entre as placas:

i.1) que tipo de ***dados*** suas placas vão trocar?

i.2) qual '***protocolo'*** será seguido durante a comunicação?

Para ajudar, algumas questões a serem respondidas:

(1) apenas começo enviar dados depois que a placa do outro '*concordar'* em receber? – *protocolo*!

(2) Ou envio dado '*direto'* sem me preocupar se está sendo recebido? - *protocolo*

(3) Envio o dado cru lido no ADC? – *formato dos dados*

(4) Ou envio o valor de cada dígito? - *formato*

(5) Como enviar o valor do cronômetro? – *formato*

Vocês decidem...

Temos tempo para elaborar o projeto e as aulas de LAB serão para dar suporte à criação do projeto. Serão apresentadas as rotinas para mostrar dados nos displays, para conversão ADC, e comunicação via UART.

Lembre-se: uma hora de planejamento elimina 10 horas de retrabalho!

*** você vai mostrar funcionando em sala de aula, e ** vai enviar o a pasta SRC do projeto zipada por e-mail.*

A nota dependerá do programa estar funcionando, da criatividade, da clareza do código, da participação e presença do aluno durante o curso.

Modos de OPERAÇÃO: (DISPLAY + AÇÕES: comunicação e leds)

1 – DRA: mostra RA, não requisita nada, não envia nada;

2 – DRA_TXL: mostra RA estático; assincronamente, (recebe Rx reqCRN), envia locCRN, (Rx reqADC), envia locADC;

3 – DRA_DREM: mostra RA (2s); (envia Tx reqCRN) recebe/mostra extCRN (2s, piscar B13); (Tx reqADC) recebe/mostra extADC (2s, piscar B12);

4 – DLOC: mostra locCRN (4s, piscar B15); mostra locADC(4s, piscar B14); não envia, não recebe;

5 – DLOC_TXL: mostra locCRN(4s, piscar B15); mostra locADC(4s, piscar B14); assincronamente recebe (Rx reqCRN) e envia locCRN, recebe (Rx reqADC) e envia locADC;

6 – DLOCREM: mostra locCRN (2s, piscar B15); mostra locADC(2s, piscar B14); (Tx reqCRN) recebe/mostra extCRN (2s, piscar B13); (Tx reqADC) recebe/mostra extADC (2s, piscar B12);

7 – DLOCREM_TXL: mostra locCRN (2s, piscar B15); mostra locADC(2s, piscar B14); (Tx reqCRN) recebe/mostra extCRN (2s, piscar B13); (Tx reqADC) recebe/mostra extADC (2s, piscar B12); assincronamente, (recebe Rx reqCRN) e envia locCRN; recebe (Rx reqADC) e envia locADC;

Modos de OPERAÇÃO: (ESTADO + INPUTS + AÇÕES: display, comunicação e leds) (depois do teste...)

1 – **DRA** [timeout rotina teste]: display RA, sem acender led algum;
- 1.1 PA1: {**DLOC**};
- 1.2 PA2: →REQSRV, {**DRAT**}; reseta $fLgERR = 0$;
- 1.3 PA3: →REQOFF {desconsidera – não muda de estado};
- 1.4 ←REQSRV {**DRA_DREM**};
- 1.5 ←REQCRN {desconsidera}; ou {**DRAT**}; *subentende que deve teve uma requisição de serviço?
- 1.6 ←REQADC {desconsidera}; ou {**DRAT**}; *subentende que deve teve uma requisição de serviço?
- 1.7 ←REQOFF {desconsidera};
- 1.8 ←MSGDND {desconsidera};

2 – **DRAT** [→REQSRV]: display RA, assincronamente, (Rx reqCRN) envia locCRN, (Rx reqADC) envia locADC; sem leds;
- 2.1 PA1: {**DLOCT**};
- 2.2 PA2: →REQSRV*, {não muda de estado}; $fLgERR = 0$; *redundante envia novamente?
- 2.3 PA3: →REQOFF {**DRA**};
- 2.4 ←REQSRV {**DRA_DREMT**};
- 2.5 ←REQCRN {não muda estado};
- 2.6 ←REQADC {não muda estado};
- 2.7 ←REQOFF {desconsidera};
- 2.8 ←MSGDND {não muda estado}; $fLgERR = 1$: * serviço solicitado, está em TxL, outro kit diz que não está mais mostrando!

3 – **DRA_DREM** [←REQSRV]: ***DRA_DREM_A:*** display RA (2s); ***DRA_DREM_B:*** (Tx reqCRN) Rx/display extCRN 2s, pisca B13; ***DRA_DREM_C:*** (Tx reqADC) Rx/display extADC 2s, piscar B12;
- 3.1 PA1: {**DLOC_DREM**};
- 3.2 PA2: →REQSRV, {**DRA_DREMT**}; $fLgERR = 0$;
- 3.3 PA3: →REQOFF {não muda estado};
- 3.4 ←REQSRV {desconsidera};
- 3.5 ←REQCRN {desconsidera}; ou {**DRA_DREMT**} *subentende que deve teve uma requisição de serviço?
- 3.6 ←REQADC {desconsidera}; ou {**DRA_DREMT**} *subentende que deve teve uma requisição de serviço?
- 3.7 ←REQOFF {**DRA**};
- 3.8 ←MSGDND {desconsidera};

4 – **DRA_DREMT** [←REQSRV]: ***DRA_DREMT_A:*** display RA (2s); ***DRA_DREMT_B:*** (Tx reqCRN) Rx/display extCRN 2s, pisca B13; ***DRA_DREMT_C:*** (Tx reqADC) Rx/display extADC 2s, piscar B12; assincronamente, recebe (Rx reqCRN) envia locCRN, recebe (Rx reqADC) envia locADC;
- 4.1 PA1: {**DLOC_DREMT**};
- 4.2 PA2: →REQSRV, {não muda de estado}; $fLgERR = 0$; *redundante envia novamente?
- 4.3 PA3: →REQOFF {**DRA_DREM**};
- 4.4 ←REQSRV {desconsidera};
- 4.5 ←REQCRN {não muda estado};
- 4.6 ←REQADC {não muda estado};
- 4.7 ←REQOFF {**DRAT**};
- 4.8 ←MSGDND {não muda estado}; $fLgERR = 1$: * serviço solicitado, está em TxL, outro kit diz que não está mais mostrando!

5 – **DLOC** [PA1]: display local; ***DLOC_A:*** (display locCRN, 4s, pisca PB15); ***DLOC_B:*** (display locADC, 4s, pisca PB14);

5.1 PA1: {desconsidera};
5.2 PA2: →REQSRV, {**DLOCT**}; $fLgERR = 0$;
5.3 PA3: →REQOFF {desconsidera};
5.4 ←REQSRV {**DLOC_DREM**};
5.5 ←REQCRN {desconsidera}; ou {**DLOCT**}; *subentende que deve teve uma requisição de serviço?
5.6 ←REQADC {desconsidera}; ou {**DLOCT**}; *subentende que deve teve uma requisição de serviço?
5.7 ←REQOFF {desconsidera};
5.8 ←MSGDND {desconsidera};

6 – **DLOCT** [→REQSRV]: ***DLOCT_A:*** (display locCRN, 4s, pisca PB15); ***DLOCT_B:*** (display locADC, 4s, pisca PB14); <u>assincronamente</u>, (Rx reqCRN) envia locCRN, (Rx reqADC) envia locADC;

6.1 PA1: {desconsidera};
6.2 PA2: →REQSRV, {não muda de estado}; $fLgERR = 0$; *redundante envia novamente?
6.3 PA3: →REQOFF {**DLOC**};
6.4 ←REQSRV {**DLOC_DREMT**};
6.5 ←REQCRN {não muda estado};
6.6 ←REQADC {não muda estado};
6.7 ←REQOFF {desconsidera};
6.8 ←MSGDND {não muda estado}; $fLgERR = 1$: * serviço solicitado, está em TxL, outro kit diz que não está mais mostrando!

7 – **DLOC_DREM** [←REQSRV]: ***DLOC_DREM_A:*** display locCRN (2s, piscar B15); ***DLOC_DREM_B:*** display locADC(2s, piscar B14); ***DLOC_DREM_C:*** (Tx reqCRN) recebe/mostra extCRN (2s, piscar B13); ***DLOC_DREM_D:*** (Tx reqADC) recebe/mostra extADC (2s, piscar B12);

7.1 PA1: {**DLOC**}; *enviar [→MSGDND] para indicar que aqui não vai mais mostrar dados remotos
7.2 PA2: →REQSRV, {**DLOC_DREMT**}; $fLgERR = 0$;
7.3 PA3: →REQOFF {não muda estado};
7.4 ←REQSRV {desconsidera};
7.5 ←REQCRN {desconsidera}; ou {**DLOC_DREMT**} *subentende que deve teve uma requisição de serviço?
7.6 ←REQADC {desconsidera}; ou {**DLOC_DREMT**} *subentende que deve teve uma requisição de serviço?
7.7 ←REQOFF {**DLOC**};
7.8 ←MSGDND {desconsidera};

8 – **DLOC_DREMT** [→REQSRV]: ***DLOC_DREMT_A:*** display locCRN (2s, piscar B15); ***DLOC_DREMT_B:*** display locCRN (2s, piscar B14); ***DLOC_DREMT_C:*** (Tx reqCRN) recebe/mostra extCRN (2s, piscar B13); ***DLOC_DREMT_D:*** (Tx reqADC) recebe/mostra extADC (2s, piscar B12); <u>assincronamente</u> ocorre (Rx reqCRN) envia locCRN, (Rx reqADC) envia locADC;

8.1 PA1: {**DLOCT**}; *enviar [→MSGDND] para indicar que aqui não vai mais mostrar dados remotos;
8.2 PA2: →REQSRV, {não muda de estado}; $fLgERR = 0$; *redundante envia novamente?
8.3 PA3: →REQOFF {**DLOC_DREM**};
8.4 ←REQSRV {desconsidera};
8.5 ←REQCRN {não muda estado};
8.6 ←REQADC {não muda estado};
8.7 ←REQOFF {**DLOCT**};
8.8 ←MSGDND {não muda estado}; $fLgERR = 1$: * serviço solicitado, está em TxL, outro kit diz que não está mais mostrando!