UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

Relatório do Lab 4

Disciplina de Sistemas Embarcados – Prof. Douglas Renaux

Autores: João Felipe Sarggin Machado e Waine Barbosa de Oliveira Junior

Versão: 26-Set-2022

1 Introdução

Durante o desenvolvimento de uma aplicação em um sistema embarcado é muito comum a interação com dispositivos I/Os. Muitas vezes esses dispositivos têm entradas e saídas relativamente simples, como um simples sinal alto ou baixo. Porém, para dispositivos mais complexos, é comum a definição e uso de interfaces mais complexas.

Para interação entre dispositivos muitas vezes é necessária a execução de diversas operações de preparo e também de controle de comunicação. Isso vai desde definir a frequência do clock de operação de um PWM até a paridade dos bits numa comunicação por *serial*.

Visando facilitar e unificar a comunicação de dispositivos para diversos fins, porém que possuem características semelhantes de comunicação, foram criadas interfaces e protocolos padrões de comunicação. Alguns exemplos de protocolos são: UART (*Universal Asynchronous Receivers/Transmitters*), I2C (*Inter-Integrated Circuit*), QSSI (*Quad Synchronous Serial Interface*), dentre outros. Algumas das características de tais protocolos são a definição de entradas e saídas padrões (como TX e RX para UART ou SCL e SDA para I2C) e padrões para comunicação serial de dados.

Além de tais protocolos, é comum em microcontroladores dispositivos para tratamento de casos de uso comuns. Como a leitura ou escrita em um pino de maneira geral, por meio de um GPIO (general purpose input/outputs), ou a geração de um PWM (pulse width modulator). Tais soluções não são complexas ao ponto da necessidade de definição de um protocolo específico para tal, porém são comuns o bastante para serem padronizadas para cada placa.

Esse laboratório tem como objetivo de aprendizado o uso das interfaces da Tiva em conjunto com a BoosterPack. Para isso são utilizados o Joystick da BoosterPack, o qual necessita de uma conversão entre um valor analógico (tensão) para um digital, portanto necessita de uma interface ADC (*analog-to-digital converter*). Também é utilizado o LED RGB, o qual possui uma operação por meio de PWM, portanto necessita da utilização do módulo de PWM da Tiva.

As seções deste relatório são organizadas como descrito a seguir: inicialmente é apresentado o planejamento das fases de desenvolvimento; então é feita a especificação do problema e da solução para tal; após isso os estudos dos recursos necessários de hardware e software; em seguida o design para implementação e as configurações da IDE IAR; por fim os testes realizados para aferir o funcionamento do sistema.

Obs.: Durante o documento, ao se referir à *Tiva*, a placa que está sendo referenciada é a *Tiva C Series TM4C1294* com processador *TM4C1294NCPDT*. Também a BoosterPack se refere à *BOOSTXL-EDUMKII Educational BoosterPack™ Plug-in Module Mark II*.

2 Planejamento das fases do processo de desenvolvimento

As atividades de desenvolvimento foram planejadas conforme a seguir:

1. Planejamento das fases do processo de desenvolvimento.
2. Definição do problema a ser resolvido.
3. Estudo da plataforma de HW (placa Tiva e seu processador).
    a. Tiva (TM4C1294NCPDT)
    b. BoosterPack (BOOSTXL-EDUMKII Educational BoosterPack™ Plug-in Module Mark II)
    c. Periféricos e protocolos de comunicação
4. Estudo da plataforma de SW
    a. TivaWare
    b. Respectivas bibliotecas para cada periférico e/ou protocolo
5. Projeto (design) da solução:

   a. Planejamento das estruturas de dados;
   b. Forma de passagem de parâmetros;
   c. Algoritmo e alocação de variáveis aos registradores.
6. Configuração do projeto na IDE (IAR).
7. Implementação da solução
8. Teste e depuração.

A documentação dos resultados foi feita conforme a realização de cada atividade.

O formato de execução das atividades foi sequencial, portanto foram feitos os estudos de todos os hardware utilizados, só então iniciado o estudo do software e bibliotecas e assim por diante.

3 Definição do problema a ser resolvido

O programa deve:

- Ler a posição corrente do Joystick (Horiz, Vert e botão)
- A cada 200 ms, informar os valores lidos pela serial UART0 (115.200 bps, 8N1)
- Ativar o LED RGB com cores relativas ao valor lido do Joystick

Restrições:

- Registre as ISRs que você criar no vetor de exceções no arquivo de inicialização. Não utilizar o mecanismo disponível no TivaWare de mapeamento do vetor de exceções em RAM, usando funções tais como *IntRegister()*.

Obs.: O objetivo de "leitura de luminosidade via sensor na BoosterPack" foi descartado, conforme dado como opção pelo professor

4 Estudo da plataforma de HW

Para a plataforma de hardware, foram feitos estudos dos dispositivos utilizados e suas comunicações com a placa Tiva. Essa seção é dividida por dispositivos

UART (*Universal Asynchronous Receivers/Transmitters*)

Os pinos RX (*receive*) e TX (*transmit*) da UART0 são mapeados aos pinos PA0 e PA1 respectivamente. É possível programar a frequência para uma banda de até 7.5 Mbps. Também há duas FIFOs, uma para o TX e outra para o RX, de 16 bytes cada para transmissão de dados.

Por padrão, a interface UART0 da Tiva é mapeada para porta COM virtual da conexão USB. Portanto, para uso dessa interface pelo computador basta utilizar a mesma porta utilizada pela conexão USB.

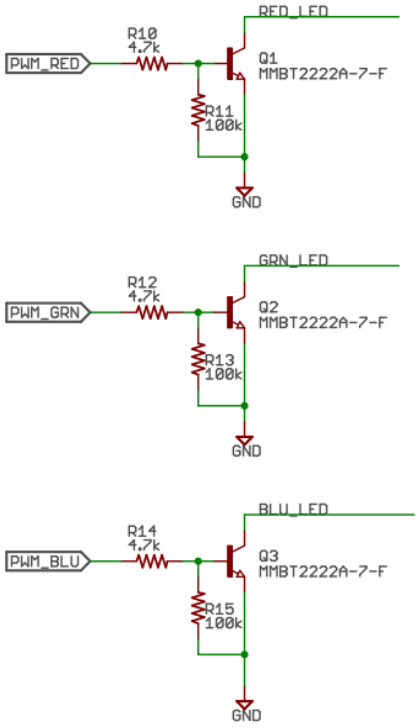
Para a correta utilização dos pinos PA0 e PA1, é necessário configurá-los para serem utilizados em sua função alternativa por meio do registrador ***GPIO Alternate Function Select (GPIOAFSEL)*** e em seguida selecionar qual das funções alternativas será utilizada, por meio do registrador ***GPIO Alternate Function Select (GPIOAFSEL)***, que nesse caso deve ser selecionado a função alternativa 0x1, representando respectivamente para os pinos PA0 e PA1, as funções ***U0Rx*** e ***U0Tx***.

LED RGB

O LED RGB é localizado na BoosterPack e possui um pino para controle da intensidade de cada cor (*red*, *green* e *blue*). Esses pinos são mapeados nos conectores conforme a tabela abaixo:

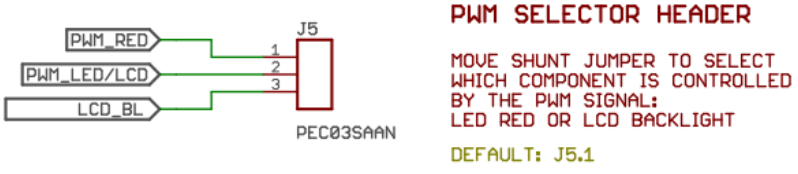
Cor	Pino BoosterPack	Pino Tiva
Red	J4.39	PF2 (M0PWM2)
Green	J4.38	PF3 (M0PWM3)
Blue	J4.37	PG0 (M0PWM4)

O controle de cada intensidade deve ser feito por meio de um PWM, com o período alto desse sendo proporcional à intensidade. O esquemático do PWM do LED na placa é mostrado abaixo.



Esquemático do controle dos LEDs RGBs

A BoosterPack também apresenta um *PWM SELECTOR HEADER* conforme descrito no manual, o qual seleciona qual saída é controlada pelo PWM. Essa saída é multiplexada entre o LCD e o PWM do Led vermelho. Então é necessário garantir que o jumper esteja conectado no pino J5.1 (o qual é sua conexão padrão).



Pino de seleção de PWM

PWM (Pulse Width Modulator)

Para uso do LED é necessário fazer uso de PWM. Na Tiva há um módulo de PWM com quatro blocos geradores um um bloco de controle, totalizando um total de 8 saídas PWM. O bloco de controle determina a polarização dos sinais e quais deles são passados aos pinos.

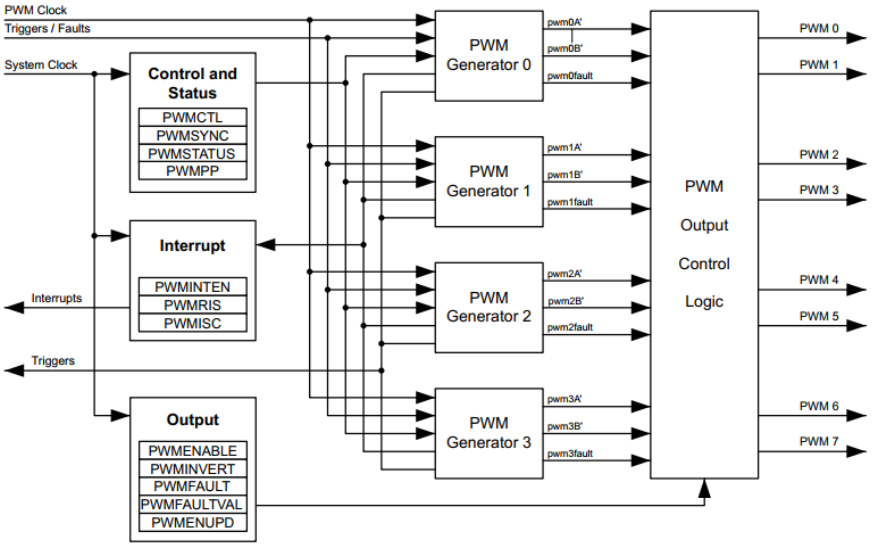


Diagrama do módulo do PWM

Os pinos mapeados para os LEDs, PF2 (M0PWM2), PF3 (M0PWM3) e PG0 (M0PWM4) são controlados respectivamente pelos geradores 1, 1 e 2. Para geração do pulso é possível utilizar como clock o próprio clock do sistema ou uma divisão desse clock.

Cada gerador possui um contador de 16 bits, que pode contar para cima ou para baixo. Também dois comparadores que produzem a saída quando os valores são iguais. Ou seja, a geração de sinais é dependente do clock e dos valores de comparação.

Para a correta utilização desses pinos, eles devem ser configurados para uso em sua função alternativa (por meio do registrador GPIOAFSEL) e em seguida selecionar a função 0x6 (por meio do registrador GPIOPCTL), representando respectivamente para os pinos PF2, PF3 e PG0, as funções M0PWM2, M0PWM3 e M0PWM4.

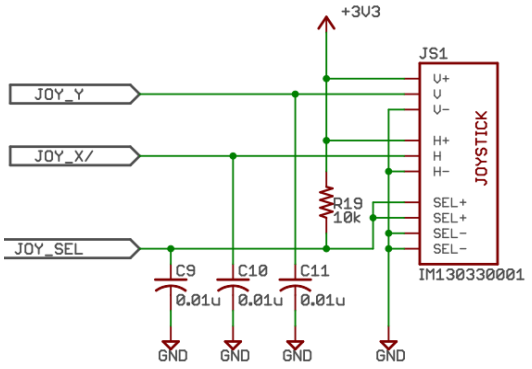
Joystick

O Joystick localiza-se na BoosterPack e possui três pinos, um para sua posição horizontal, outro para sua posição vertical e um terceiro para checagem do valor de seleção (clicar). Esses pinos são mapeados conforme a tabela abaixo

Sinal	Pino BoosterPack	Pino Tiva
Horizontal	J1.2	PE4 (AIN9)
Vertical	J3.26	PE3 (AIN0)
Seleção	J1.5	PC6

O dispositivo é um IM130330001, o qual consiste em dois potenciômetros (para movimento horizontal e vertical) e um botão (para clicar) que é alto quando pressionado. Pelo fato do dispositivo gerar uma tensão conforme o movimento dos potenciômetros, é necessário transformar o sinal analógico em digital. Por exemplo, com o botão para esquerda, se lê X=0.

O esquemático com as conexões do joystick segue abaixo



Esquemático do Joystick na BoosterPack

ADC (*Analog-to-digital converter*)

Tendo em vista que o joystick gera uma tensão variável, a qual indica a posição do analógico, é necessário utilizar um conversor desse sinal para o sinal digital, que é o ADC. A Tiva possui dois ADCs que convertem uma tensão analógica para um valor digital. O bloco de diagrama do módulo ADC é apresentado abaixo

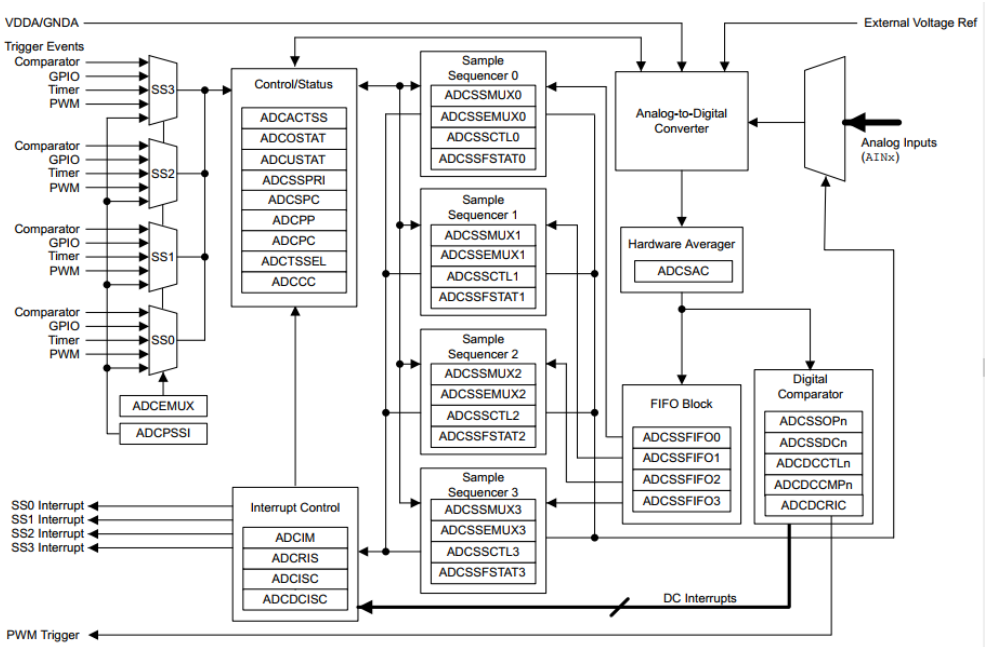


Diagrama de blocos do ADC

Cada módulo ADC possui 12 bits de precisão e os dois compartilham 20 canais de entrada. O *trigger* pode ser programado por software, timers, PWM ou GPIO. Os pinos PE3 e PE4, utilizados pelo analógico, são dois dos 20 pinos de entrada dos módulos ADCs.

Os dois módulos operam de maneira independente. Portanto é possível utilizar um para ler a posição vertical e outro para ler a posição horizontal do analógico.

O ADC possui 4 FIFOs (SS0-SS3) para amostragem (*sampling*) das leituras, com a SS0 tendo a maior capacidade (8 amostras). Cada amostra é uma *word* de 32-bits, com valor da conversão nos 12 bits menos significativos.

Para a correta utilização do pino PC6 (Ligado ao *Select* do *Joystick*) ele deve ser configurado como pino de entrada digital, por meio dos registradores ***GPIO Digital Enable (GPIODEN)*** e ***GPIO Direction (GPIODIR)***. Não é necessário configurar um resistor de ***pull-up*** pois a BoosterPack já o inclui na própria placa, conforme mostra o esquemático do ***Joystick***.

Já para os pinos PE4 e PE3, não são necessárias configurações adicionais, pois por padrão já são entradas analógicas, ainda sim, há funções específicas na TivaWare que garantem a configuração desses pinos nesse modo (GPIOPinTypeADC).

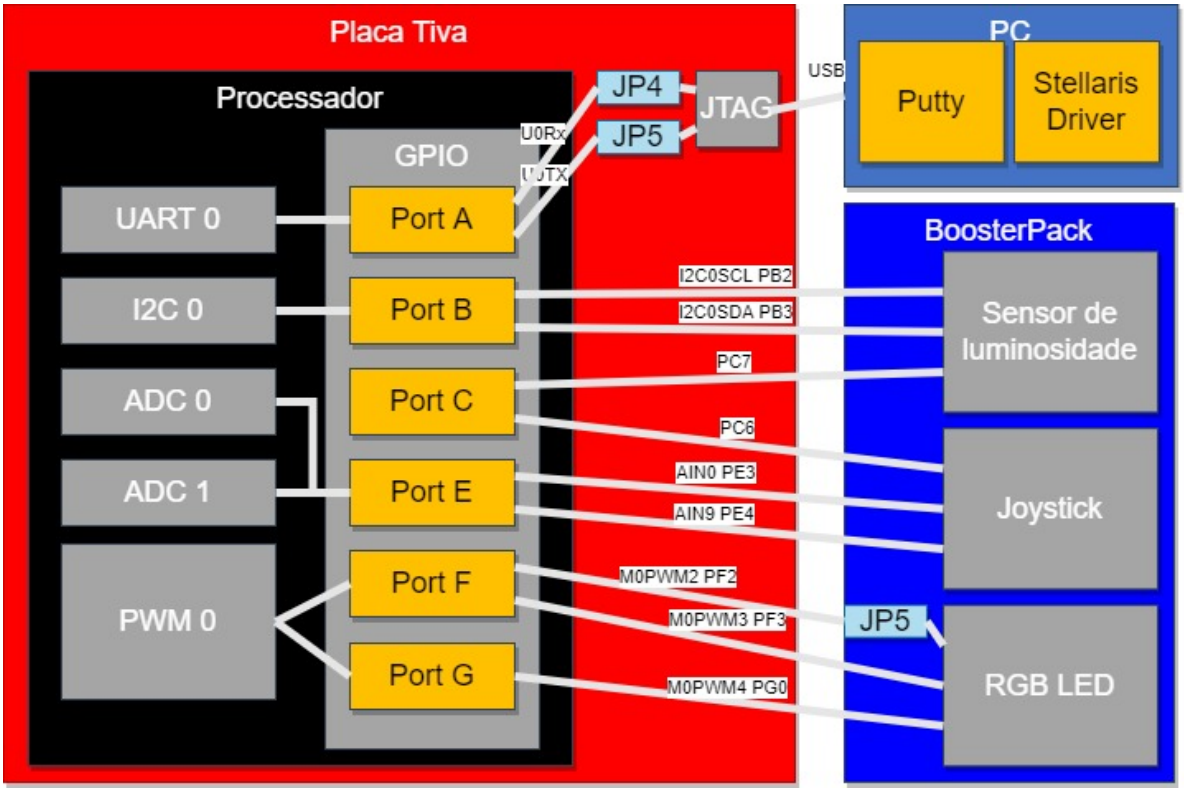


Diagrama de blocos do hardware

5 Estudo da plataforma de SW

Para atualizar a frequência do processador e ativar periféricos será necessário utilizar funções relativas ao Sistema de Controle, são elas:

SysCtlClockFreqSet(config, sysclock): Configura a frequência do sistema.

SysCtlPeripheralEnable(peripheral): Ativa um periférico.

Já no módulo GPIO, será necessário configurar pinos para utilizarem as funções alternativas e para habilitar e configurar interrupções para o botão de seleção do joystick e para o pino de interrupção do sensor de luminosidade:

GPIOPinConfigure(pinconfig): Configura uma função alternativa para o pino.

GPIOPinTypePWM(port, pin): Configura um pino para ser utilizado pelo módulo PWM.

GPIOPinTypeADC(port, pin): Configura um pino para ser utilizado como entrada do conversor analógico-digital.

GPIOPinTypeGPIOInput(port, pin): Configura um pino para ser utilizado como entrada digital.

GPIOIntTypeSet(port, pin, inttype): Configura uma interrupção de um determinado tipo para um pino.

GPIOIntEnable(port, intflags): Ativa as interrupções para um pino a nível de periférico.

Como serão utilizadas interrupções, será necessário ativá-las no controlador NVIC, utilizando para isso a seguinte função:

IntEnable(interrupt): Ativa uma interrupção a nível NVIC.

Para configurar o LED RGB, será necessário configurar o módulo PWM:

PWMClockSet(base, config): Configura a frequência que será utilizada no módulo PWM.

PWMGenConfigure(base, gen, config): Configura um gerador PWM.

PWMGenPeriodSet(base, gen, period): Define o período de contagem em número de ticks para um gerador PWM.

PWMPulseWidthSet(base, pwmout, width): Configura a porção do período em que a saída terá nível lógico alto em número de ticks para um gerador PWM e também configura em qual dos pinos disponíveis a saída será disponibilizada.

PWMOutputState(base, pwmoutbit, enable): Habilita ou desabilita a saída PWM para um determinado pino.

Para ler o estado do *joystick* será necessário configurar os conversores analógico-digital, utilizando para isso as seguintes funções:

ADCSequenceConfigure(base, sequencenum, trigger, priority): Configura uma fonte de gatilho de amostragem e uma prioridade para um amostrador.

ADCPhaseDelaySet(base, phase): Configura um atraso em ticks entre o gatilho e o início da amostragem para um amostrador.

ADCSequenceStepConfigure(base, sequencenum, step, config): Configura como será feita a amostragem, incluindo a janela de tempo, o pino de entrada do sinal analógico e quantas amostragens serão feitas por ciclo.

ADCHardwareOversampleConfigure(base, factor): Coleta várias amostras para tirar uma média dado um determinado fator.

ADCSequenceEnable(base, sequencenum): Habilita o funcionamento de um amostrador.

ADCIntEnable(base, sequencenum): Habilita as interrupções para um determinado amostrador.

Para controlar a temporização do envio das informações pela porta serial, será utilizado um temporizador, onde as seguintes funções são relevantes:

TimerConfigure(base, config): Configura um temporizador.

TimerLoadSet(base, timer, value): Carrega um valor de contagem em ticks para um temporizador.

TimerIntEnable(base, intflags): Habilita um tipo de interrupção para um temporizador.

TimerEnable(base, timer): Inicia a contagem de um temporizador.

Quanto o envio dos dados lidos pela porta serial, será necessário a utilização das seguintes funções:

UARTClockSourceSet(base, source): Define a fonte do clock da porta serial.

UARTConfigSetExpClk(base, uartclk, baud, config): Define a frequência de operação e a configuração dos pacotes para uma determinada porta serial.

UARTFIFOEnable(base): Habilita o uso de uma fila de transmissão e recepção para a porta serial.

UARTEnable(base): Inicia o funcionamento da porta serial.

Separando essas funções em 5 tipos, pode-se definir a ordem correta de suas execuções:

1°) Funções que ativam os periféricos e as portas GPIO.

2°) Funções que configuram os periféricos.

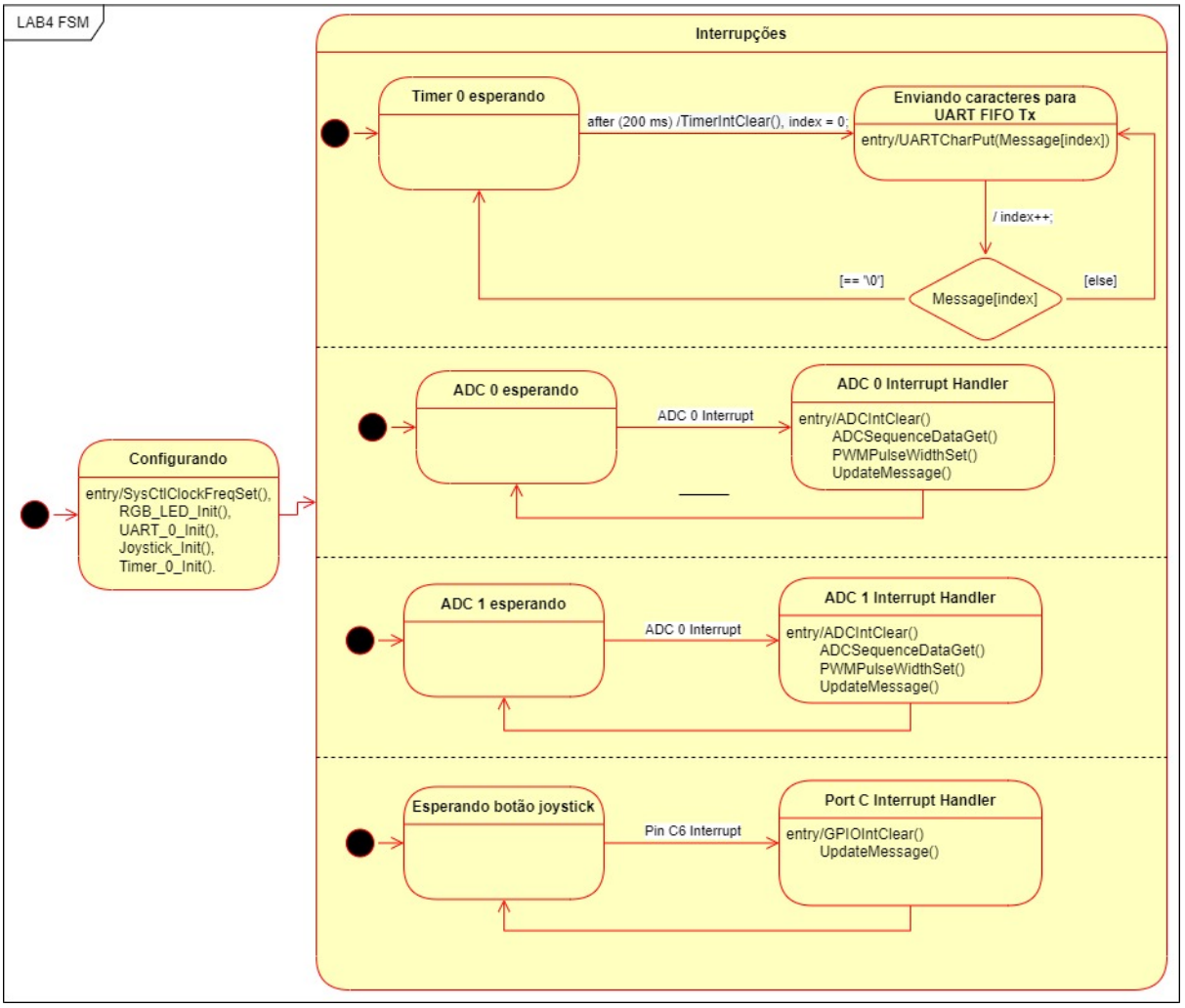
3°) Funções que configuram os pinos GPIO.

4°) Funções que configuram e habilitam as interrupções.

5°) Função que ativam o funcionamento do periférico.

6 Projeto (design) da solução

A solução irá se comportar conforme a seguinte máquina de estados:



Máquina de estados da solução

7 Configuração do projeto na IDE (IAR).

As configurações básicas incluem a seleção do dispositivo "Texas Instruments TM4C1294NCPDT", o "VFPv4 single precision" como versão da unidade de ponto flutuante (FPU) e a configuração compacta ("Normal") da biblioteca de tempo de execução C/C++14 (onde uma ponto importante a se observar é a falta de suporte à descritor de arquivo, o que pode causar problemas na chamada da função printf dentro das interrupções, algo que naturalmente não é recomendado, mas com um suporte completo - "Full" - poderia ser feito).

As formatações das funções "printf" e "scanf" estão no modo automático (compilador irá ajustar conforme necessidade encontrada). A seleção de Heap também está no modo automático.

Já o compilador está definido para a linguagem C, com o dialeto Standard C e em conformidade com a norma padrão com extensões do IAR. A otimização está definida para o nível máximo, priorizando o tamanho do código de saída.

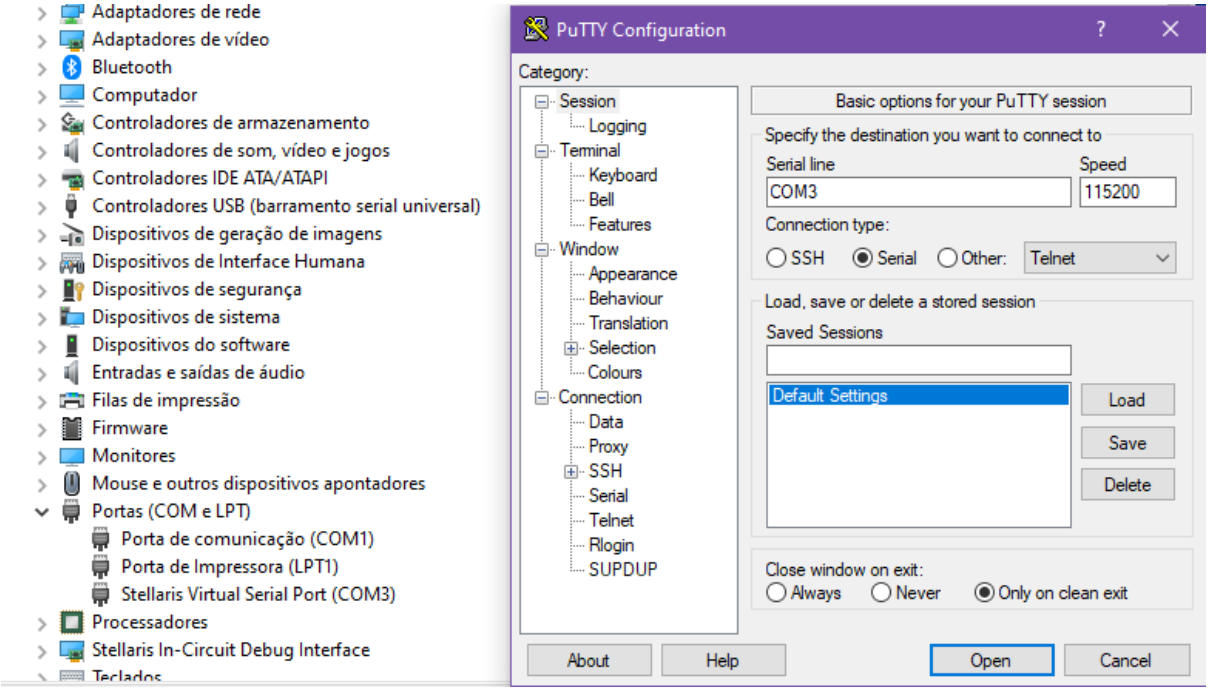
8 Teste e depuração.

Os testes serão divididos por funcionalidade, testando primeiro um “Hello World” para cada uma e então aumentando gradualmente a complexidade até que a aplicação estivesse completa.

Aqui serão apresentados os testes para cada componente e então os testes finais de integração.

UART

Para a comunicação com a UART, será utilizado o software *Putty* e configurada as entradas conforme especificado (115200 bits per second, 8N1 e a entrada COM3 nesse caso). Caso haja necessidade de enviar dados também, é necessário colocar em “Force On” as opções "Local echo" e "Line editing" na aba "Terminal".



Configurações do software *Putty*

Será utilizado o projeto “*examples\boards\ek-tm4c1294xl\uart_echo*” do TivaWare para verificar o funcionamento correto da interface UART. Após isso, o código exemplo será usado como base para o setup da UART no projeto e então testado com a função *UARTprintf* (uma função definida na pasta *utils* do TivaWare), a qual se espera que os dados sejam impressos no terminal do IAR.

LEDs RGB (PWM)

Para o teste do LED RGB, será configurado o pino PF2 como saída 2 do módulo PWM.

Tal saída é gerenciada pelo gerador 1 do módulo PWM, portanto o mesmo será configurado para um período de 256 ticks de clock (Para se assimilar a famosa codificação de cores RGB em 8 bits por canal) e em seguida ajustado uma largura de 255 ticks, que nesse caso representa um *Duty Cycle* de 100%.

Espera-se que o LED RGB apresente uma coloração vermelha extremamente nítida, pois se trata da ativação total deste canal.

Joystick (ADC)

Para o teste do *joystick*, será configurado primeiramente o pino PC6 como entrada digital.

Uma interrupção será associada a esse pino por borda de descida e no tratamento dessa interrupção será aceso o LED de usuário 1 da placa Tiva. Espera-se que ao pressionar o botão de seleção do *joystick* o LED 1 seja aceso.

Já para o teste dos eixos, será configurado o pino PE4 como entrada de um dos conversores analógico-digital e escrito uma rotina de tratamento de interrupção para ele. Nesta rotina será impresso no terminal do IAR o valor da conversão e espera uma mudança nos valores conforme o *joystick* é movimentado no eixo horizontal.

Integração

O teste de integração se dará por meio da simples inspeção visual dos valores impressos no Putty. Espera-se que ao deixar o *joystick* na posição de repouso, tanto o valor horizontal quanto o vertical estejam próximos de 2048 (que corresponde a metade da escala de conversão de 12 bits). Analogamente, ao se mover o *joystick* para a direita ou para cima, espera-se a visualização de valores próximos a 4095 respectivamente no eixo horizontal e vertical, ou os valores próximos a zero nos mesmo eixos caso o *joystick* seja movido para a esquerda ou para baixo. Se o botão de seleção for pressionado, espera-se uma mudança no valor de zero para um.