Universidade Federal de Pernambuco
Departamento de Engenharia Mecânica

Laboratório de Sistemas Digitais — 1º Semestre 2019
Prof. João Paulo Cerquinho Cajueiro — 10 de maio de 2019

Aluno: _____

Experimento III – Data Logger

Objetivo – Construir um data logger para os dados do acelerômetro presente na FRDM-KL25Z.

Especificação

A partir da placa FRDM-KL25Z, construir um data logger.

O data logger fará uma medida de aceleração nos 3 eixos a cada x décimos de segundo, onde x fica entre 1 e 128 (12,8 s).

A cada 1 s o data logger deve piscar o led por 0.1 s. Verde se tudo ok. Amarelo se memória ficando cheia ($> 90\%$), azul se desligado e vermelho se com problema.

Uma comunicação serial serve para controlar e pegar os dados do data logger. O data logger sempre recebe 3 bytes:

INICIO	COMANDO	FIM

onde `INICIO` é um byte com valor 0x55, `FIM` tem valor 0xAA e `COMANDO` pode ser:

000 0001 - ping.

0000 0010 - status.

0000 0011 - erase.

0000 0100 - start.

0000 0101 - stop.

0000 0110 - lastVal.

0000 0110 - dump.

1nnn nnnn - setVel.

1 Respostas aos comandos

Cada comando tem uma resposta específica.

1.1 Ping

O ping faz apenas com que o data logger retorne um pacote | INICIO | 0x11 | FIM |, onde INICIO e FIM tem os mesmos valores dos comandos. Chamaremos este pacote de `ACK`.

1.2 Status

O comando de status faz o datalogger retornar os seguintes bytes:

INICIO	STATUS	VEL	MEM	FIM

O byte `STATUS` tem os seguintes campos:

bit 0 - 1 se executando, 0 se parado.

bit 1 - 1 se memória maior que 90%, 0 caso contrário.

bit 2 - 1 se memória cheia, 0 caso contrário.

demais bits - valor 0.

`VEL` mostra a taxa de aquisição programada, de 0 (1 medida a cada 100 ms) a 127 (1 medida a cada 12,8 s). `MEM` mostra a quantidade de memória já utilizada: 0 – 0% a 255 – 100%.

1.3 Erase

O comando erase só é aceito quando o datalogger esá parado ou logo em seguida a um dump. Ele faz com que o ponteiro da memória volte à posição 0. Caso aceito, deve retornar um `ACK`. Caso seja recebido com o sistema executando e sem ser após um dump, deve retornar um `NAK`: | INICIO | 0xEE | FIM |.

1.4 Start

Se o sistema estiver parado, deve iniciar o sistema e retornar um `ACK`. Caso o sistema já esteja executando, deve apenas retornar `ACK`. Caso não possa iniciar o sistema (memória cheia, falha de comunicação com o acelerômetro), deve retornar `NAK`.

1.5 Stop

Para o sistema se estiver executando e retorna um `ACK`.

1.6 LastVal

Deve retornar o último valor medido, no formato:

INICIO	IDX_MSB	IDX_LSB	X_MSB	X_LSB	Y_MSB	Y_LSB	Z_MSB	Z_LSB	CRC	FIM

`IDX_MSB` e `IDX_LSB` é simplesmente o índice da medida desde a inicialização ou do último comando erase, dividido em byte mais significativo (MSB) e menos significativo (LSB). Os bytes `X_MSB`, `X_LSB`, `Y_MSB`, `Y_LSB`, `Z_MSB` e `Z_LSB` correspondem às acelerações nos eixos x, y e z, com 14 bits, divididos em byte mais significativo (MSB) e menos significativo (LSB).

`CRC` é o ou exclusivo dos bytes anteriores:

$$\text{CRC} = \text{IDX\_MSB} \oplus \text{IDX\_LSB} \oplus \text{X\_MSB} \oplus \text{X\_LSB} \oplus \text{Y\_MSB} \oplus \text{Y\_LSB} \oplus \text{Z\_MSB} \oplus \text{Z\_LSB}$$

1.7 Dump

O comando dump faz com que o datalogger retorne todos os valores medidos até então. Ele faz isso mandando um pacote de início de dump: | INICIO | 0x55 | N_MSB | N_LSB | FIM |, seguido dos N pacotes de medidas, iguais ao da resposta ao comando LastVal, após os quais deve mandar um pacote terminador: | INICIO | 0xAA | FIM |.

1.8 SetVel

Caso o sistema esteja parado, este comando ajusta o período entre aquisições para o valor dado pelos bits nnnnnnn, de 0 (1 medida a cada 100 ms) a 127 (1 medida a cada 12,8 s) e retorna um ACK. Caso o sistema esteja rodando, deve apenas retornar um NAK.

2 Primeira parte – Comunicação com acelerômetro

O acelerômetro presente na própria placa se comunica por I2C. O acelerômetro é o MMA8451Q. Duas opções para o I2C: https://github.com/evandro-teixeira/Library-FRDM-KL25Z ou o CMSIS.

Datasheet: https://cache.freescale.com/files/sensors/doc/data_sheet/MMA8451Q.pdf
App. Note: https://cache.freescale.com/files/sensors/doc/app_note/AN4076.pdf
A cada intervalo a comunicação com o acelerômetro vai:

1. Colocar o acelerômetro em modo ativo: CTRL_REG1 (0x2A) bit 0 (ACTIVE) setado.
2. Esperar que tenha dado nos três canais: Checar o STATUS (0x00) bit 3 (ZYXDR).
3. Ler os registradores de dados: 0x01 OUT_X_MSB, 0x02 OUT_X_LSB, 0x03 OUT_Y_MSB, 0x04 OUT_Y_LSB, 0x05 OUT_Z_MSB e 0x06 OUT_Z_LSB
4. Colocar o acelerômetro em standby: CTRL_REG1 (0x2A) bit 0 (ACTIVE) resetado.

3 Segunda parte – Periodização

Para gerar a interrupção a cada intervalo especificado, é interessante usar o PIT (Periodic Interrupt Timer). O PIT tem um contador de 32 bits, o que permite um período máximo de aproximadamente $\frac{2^{32}}{2 \cdot 10^7} = 214,7$ segundos.

Como há 2 canais, podemos usar um para a medida e outro para o led. Porém é difícil usar o PIT para gerar o 0,1 s. Pode-se ao disparar o TIP, ligar o led e um TPM. Ao disparar o TPM, desliga-se o TPM e o led.

4 Terceira parte – Memória flash

A memória flash é organizada em setores de 1 kByte. Só é possível escrever num setor apagado e só é possível apagar um setor por vez, embora seja possível

escrever parte por parte. Deve-se tomar cuidado para não apagar ou reescrever um setor que tenha programa.

Neste código, vamos usar 4 setores, que serão apagados ao se executar um erase. A cada medida vamos escrever 6 bytes (`X_MSB`, `X_LSB`, `Y_MSB`, `Y_LSB`, `Z_MSB` e `Z_LSB`), logo teremos espaço para 682 medidas. Tendo este valor definimos também que cada incremento no campo \verbMEM| equivale a 2 e 2/3 medidas.

5 Terceira parte - comunicação

6 Quarta parte – Modo de baixo consumo

Devemos desligar o microcontrolador entre cada medida.