**CXP – Programa Caixa Preta (04/08/2020)**

São duas as opções principais: Operação e Teste.

**Operação**: Ao ligar, entra normalmente em operação. Permite escolher um tipo de Operação para a Caixa Preta.

**Teste**: Entra em Teste se ligar a Caixa Preta com a tecla SEL acionada. Permite escolher rotinas para testar os diversos dispositivos da Caixa Preta e fazer novos ensaios.

Tudo pode ser comandado pela porta serial (eu espero!). Os comandos são letras (maiúsculas ou minúsculas) e números. O terminal do Arduino só os envia após o <Enter>.

**Comandos seriais**

|  |  |  |
| --- | --- | --- |
| Modo | Letra |  |
| - | T | Vai para o modo Teste |
| - | O | Vai para o modo Opera |
| - | Número | Inicia a opção com esse número |
| Opera 1:  Adquirir dados | S | Inicia a aquisição (corresponde ao botão SEL) |
| I | Interrompe a aquisição (corresponde ao INF) |
| X | Sai do modo aquisição |

----------------------- Distribuição da SRAM de 256 KB ---------------------------------

SRAM = 256 KB = 262.144

Mensagem GPS = 128 Bytes

Mensagem MPU = 18 bytes

Configuração = 416 bytes

Qtd msg MPU = 12.720 🡪 12.720 \* 18 = 228.960 Bytes (127,2 segundos)

Qtd msg GPS = 254 🡪 254 \* 128 = 32.768 Bytes (256 segundos)

262.144 – (256+32.768 + 228.960) = 160 bytes sobrando

Mapa da SRAM:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Finalidade** | **Faixa Hexa** | **Bytes** | **Qtd msg** | **Tempo** |
| MPU | 0 0000 🡪 3 7E5F | 228.960 | 12.720 | 127 seg |
| Configuração | 3 7E60 🡪 3 7FFF | 416 | - | - |
| GPS | 3 8000 🡪 3 FFFF | 32.768 | 256 | 256 seg |

#define MPU\_ADR\_INI 0x00000L

#define MPU\_ADR\_FIM 0x37E60L

#define CXP\_ADR\_INI 0x37E60L

#define CXP\_ADR\_FIM 0x38000L

#define GPS\_ADR\_INI 0x38000L

#define GPS\_ADR\_FIM 0x40000L

Unificação das Portas Seriais UART0 e UART2

UART0 --> Arduino

UART1 -->

UART2 --> Bluetooth

UART3 --> GPS

A CXP enxerga somente um canal serial.

**Chegada**: Não importa se vem de COM0 ou COM2, o dado é colocado na mesma fila.

**Saída**: toda vez que enviar um dado, ele sai pelas COM0 e COM2.

|  |  |
| --- | --- |
|  |  |
| byte seri\_fila[SERI\_FILA\_TAM]  byte seri\_pin,seri\_pout  byte seri\_parou;  void seri\_config(void);  byte seri\_poe(byte dado)  byte seri\_tira (byte &dado)  void seri\_cheia(void) | byte sero\_fila[SERO\_FILA\_TAM]  byte sero\_pin,sero\_pout  byte sero\_parou;  void sero\_config();  byte sero\_poe(byte dado)  byte sero\_tira (byte &dado)  void sero\_cheia(void) |

--- SERI ---

Na chegada, interrupção da UART0 ou da UART2 chama **seri\_poe()** e põe na fila. Não importa de onde veio.

Na saída, CXP usa **seri\_tira()**, que deve desabilitar interrupções para tirar da fila.

--- SERO ---

Na chegada, CXP chama **sero\_poe()**, para por algo na fila de saída (desabilitar interrupções para por na fila).

Na saída, Timer sempre reinicia a transmissão que acontece por interrupção até a fila esvaziar.

Interrupção TX UART0:

sero\_cont++;

if (sero\_cont==2){

if (sero\_tira(&x) == TRUE){

UDR0=x; UDR2=x;

sero\_parou=FALSE ; sero\_cont=0;

}

else Sero\_parou=TRUE;

}

Interrupção TX UART2:

Sero\_cont++;

if (sero\_cont==2){

if (sero\_tira(&x) == TRUE){

UDR0=x; UDR2=x;

Sero\_parou=FALSE; Sero\_cont=0;

}

else sero\_parou=TRUE;

}

Note que são duas unidades de saída UDR0 e UDR2. O dado a ser enviado é escrito em ambas. Um novo dado só pode ser enviado se ambas terminaram o envio anterior. A variável sero\_cont garante isso, ela é zerada cada vez que um dado é escrito em UDR0 e UDR2. A primeira que termina incrementa sero\_cont, que vai para 1. A segunda, ao terminar incrementa sero\_cont, que vai para 2 e é autorizada a escrever novo dado a ser transmitido e zera a variável.

Quanto a fila sero fica vazia, sero\_tira(&x) == FALSE então a transmissão deve parar por falta de dado. Mas faz a variável sero\_parou=TRUE . O timer, se encontra a sinalização de que SERO parou, verifica se tem novo dado para retirar da fila, caso positivo, retira o dado e o escreve em UDR0 e UDR2, faz sero\_parou=FALSE.

TIMER:

if sero\_parou == TRUE{

if (sero\_tira(&x) == TRUE){

UDR0=x; UDR2=x;

Sero\_parou=FALSE Sero\_cont=0;

}

else{

Sero\_parou=TRUE; //Não precisa dessa linha

}

}

**CALIBRAÇÃO DE FÁBRICA (CF)**

É armazenada na EEPROM = 4 KB = 4.069 do processador AVR

Por **Calibração de Fábrica** se entendem parâmetros importantes para o funcionamento e que o usuário não vai alterar.

Uma calibração importante é a do erro intrínseco do MPU. Vamos deixá-lo ligado por um longo tempo (10 minutos) em um local estável, perfeitamente na horizontal, e depois fazer 1024 medidas espaçadas de acordo com a frequência da amostragem, calcular o erro intrínseco de cada eixo. Será importante armazenar o local onde foi feita a calibração e a aceleração da gravidade neste local.

Tabela na EEPROM com os dados da Calibração de Fábrica

(Todas strings são terminadas com zero ‘\0’)

(Valores de 16 ou 32 bits, grava-se primeiro o MSB “Big Endian”)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Bytes** | **Tipo** | **Exemplo** | **Nome** | **Descrição** |
|  |  |  |  | **Local e aceleração da gravidade** |
| 2 | 16 bits | SS | CF\_OK | SS = já fez calibração, do contrário não |
| 14 | string | dd/mm/aa | CF\_DATA | Data da configuração |
| 32 | string | Brasilia | CF\_LOCAL | Local da configuração |
| 16 | string | 9,80665 | CFG\_PADRAO | Ac. da gravidade padrão (1g) |
| 16 | string | 9,7808439 | CFG\_LOCAL | Ac. da gravidade (m/s2) no local da configuração |
| 2 | 16 bits | 0x3FFF | CFG\_PADRAO\_BIN | Ac. da gravidade padrão na escala de +/- 2g do MPU |
| 2 | 16 bits | 0x3FD5 | CFG\_LOCAL\_BIN | Ac. da gravidade local na escala de +/- 2g do MPU |
| 2 | 16 bits | 0x75 | CF\_WHO | Resposta ao Who am I |
| - | - | - | - | **Médias e Parâmetros usados** |
| 2 | 16 bits | 100 | CF\_FA | Freq de amostragem usada (110, 200, ..., 1000) |
| 2 | 16 bits |  | CF\_BW | Banda passante do filtro (5, 10, 21, ..., 260 Hz) |
| 2 | 16 bits |  | CF\_ESC\_AC | Escala usada para o Acelerômetro (2, 4, 8, 16) |
| 2 | 16 bits |  | CF\_ESC\_GI | Escala usada p/ o Giroscópio (250, 500, 1000, 2000) |
| 2 | 16 bits |  | CF\_QTD | Quantidade de medidas para calcular a média |
| 2 | 16 bits |  | CF\_AX | Erro do eixo ax (média) |
| 2 | 16 bits |  | CF\_AY | Erro do eixo ay (média) |
| 2 | 16 bits |  | CF\_AZ | Erro do eixo az (média) – ? descontado g local? |
| 2 | 16 bits |  | CF\_TP | Temperatura média durante o cálculo |
| 2 | 16 bits |  | CF\_GX | Erro do eixo gx (média) |
| 2 | 16 bits |  | CF\_GY | Erro do eixo gy (média) |
| 2 | 16 bits |  | CF\_GZ | Erro do eixo gz (média) |
| - | - | - | - | **Somatórios** |
| 4 | 32 bits |  | CF\_AX\_SOMA | Somatório no eixo ax |
| 4 | 32 bits |  | CF\_AY\_SOMA | Somatório no eixo ay |
| 4 | 32 bits |  | CF\_AZ\_SOMA | Somatório no eixo az |
| 4 | 32 bits |  | CF\_TP\_SOMA | Somatório das temperaturas |
| 4 | 32 bits |  | CF\_GX\_SOMA | Somatório no eixo gx |
| 4 | 32 bits |  | CF\_GY\_SOMA | Somatório no eixo gy |
| 4 | 32 bits |  | CF\_GZ\_SOMA | Somatório no eixo gz |
| - | - | - | - | **Primeira e última medidas** |
| 2 | 16 bits |  | CF\_AX\_PRI | Primeira medida no eixo ax |
| 2 | 16 bits |  | CF\_AY\_PRI | Primeira medida no eixo ax |
| 2 | 16 bits |  | CF\_AZ\_ PRI | Primeira medida no eixo ax |
| 2 | 16 bits |  | CF\_TP\_ PRI | Primeira medida de temperatura |
| 2 | 16 bits |  | CF\_GX\_ PRI | Primeira medida no eixo gx |
| 2 | 16 bits |  | CF\_GY\_ PRI | Primeira medida no eixo gy |
| 2 | 16 bits |  | CF\_GZ\_ PRI | Primeira medida no eixo gz |
|  |  |  |  |  |
| 2 | 16 bits |  | CF\_AX\_ULT | Última medida no eixo ax |
| 2 | 16 bits |  | CF\_AY\_ULT | Última medida no eixo ay |
| 2 | 16 bits |  | CF\_AZ\_ ULT | Última medida no eixo ax |
| 2 | 16 bits |  | CF\_TP\_ ULT | Última medida de temperatura |
| 2 | 16 bits |  | CF\_GX\_ ULT | Última medida no eixo gx |
| 2 | 16 bits |  | CF\_GY\_ ULT | Última medida no eixo gy |
| 2 | 16 bits |  | CF\_GZ\_ ULT | Última medida no eixo gz |
| - | - | - | - | **Self Test** |
| 2 | 16 bits |  | CF\_ST\_OK | Passou no Self-test? (TRUE=OK e FALSE=NOK) |
| 2 | 16 bits |  | CF\_ST\_OFF\_AX | Leitura do eixo ax com Self Test desligado |
| 2 | 16 bits |  | CF\_ST\_OFF\_AY | Leitura do eixo ay com Self Test desligado |
| 2 | 16 bits |  | CF\_ST\_OFF\_AZ | Leitura do eixo az com Self Test desligado |
| 2 | 16 bits |  | CF\_ST\_OFF\_GX | Leitura do eixo gx com Self Test desligado |
| 2 | 16 bits |  | CF\_ST\_OFF\_GY | Leitura do eixo gy com Self Test desligado |
| 2 | 16 bits |  | CF\_ST\_OFF\_GZ | Leitura do eixo gz com Self Test desligado |
|  |  |  |  |  |
| 2 | 16 bits |  | CF\_ST\_ON\_AX | Leitura do eixo ax com Self Test ligado |
| 2 | 16 bits |  | CF\_ST\_ON\_AY | Leitura do eixo ay com Self Test ligado |
| 2 | 16 bits |  | CF\_ST\_ON\_AZ | Leitura do eixo az com Self Test ligado |
| 2 | 16 bits |  | CF\_ST\_ON\_GX | Leitura do eixo gx com Self Test ligado |
| 2 | 16 bits |  | CF\_ST\_ON\_GY | Leitura do eixo gy com Self Test ligado |
| 2 | 16 bits |  | CF\_ST\_ON\_GZ | Leitura do eixo gz com Self Test ligado |
|  |  |  |  |  |
| 2 | 16 bits |  | CF\_ST\_REG\_AX | Leitura do Reg de Self Test para ax (8 bits) |
| 2 | 16 bits |  | CF\_ST\_REG\_AY | Leitura do Reg de Self Test para ay (8 bits) |
| 2 | 16 bits |  | CF\_ST\_REG\_AZ | Leitura do Reg de Self Test para az (8 bits) |
| 2 | 16 bits |  | CF\_ST\_REG\_GX | Leitura do Reg de Self Test para gx (8 bits) |
| 2 | 16 bits |  | CF\_ST\_REG\_GY | Leitura do Reg de Self Test para gy (8 bits) |
| 2 | 16 bits |  | CF\_ST\_REG\_GZ | Leitura do Reg de Self Test para gz (8 bits) |
|  |  |  |  |  |
| 2 | 16 bits |  | CF\_ST\_TOL\_AX | Resultado calibração ax, tolerância de 14% |
| 2 | 16 bits |  | CF\_ST\_ TOL \_AY | Resultado calibração ay, tolerância de 14% |
| 2 | 16 bits |  | CF\_ST\_ TOL \_AZ | Resultado calibração az, tolerância de 14% |
| 2 | 16 bits |  | CF\_ST\_ TOL \_GX | Resultado calibração gx, tolerância de 14% |
| 2 | 16 bits |  | CF\_ST\_ TOL \_GY | Resultado calibração gy, tolerância de 14% |
| 2 | 16 bits |  | CF\_ST\_ TOL \_GZ | Resultado calibração gz, tolerância de 14% |
| - | - | - | - | **Magnetômetro – Calibração** |
| 2 | 16 bits | SS | CF\_MAG\_OK | SS = já fez calibração do magnetômetro |
| 2 | 16 bits |  | CF\_STH\_OK | Mag Passou no Self-test? (TRUE=OK e FALSE=NOK) |
| 2 | 16 bits |  | CF\_STH\_HX | Leitura de hx durante Self Test |
| 2 | 16 bits |  | CF\_STH\_HY | Leitura de hy durante Self Test |
| 2 | 16 bits |  | CF\_STH\_HZ | Leitura de hz durante Self Test |
| 2 | 16 bits |  | CF\_HX\_ASA | ASA de hx, Fuse ROM adjustment sensibility |
| 2 | 16 bits |  | CF\_HY\_ASA | ASA de hy, Fuse ROM adjustment sensibility |
| 2 | 16 bits |  | CF\_HZ\_ASA | ASA de hz, Fuse ROM adjustment sensibility |
| 2 | 16 bits |  | CF\_HX\_OFF | Offset de hx (dividir por 10) Hard Iron |
| 2 | 16 bits |  | CF\_HY\_OFF | Offset de hy (dividir por 10) Hard Iron |
| 2 | 16 bits |  | CF\_HZ\_OFF | Offset de hz (dividir por 10) Hard Iron |
| 2 | 16 bits |  | CF\_HX\_ESC | Escala de hx (dividir por 10) Soft Iron |
| 2 | 16 bits |  | CF\_HY\_ESC | Escala de hy (dividir por 10) Soft Iron |
| 2 | 16 bits |  | CF\_HZ\_ESC | Escala de hz (dividir por 10) Soft Iron |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

Note que as medidas CF\_ST\_REG\_AX, ..., CF\_ST\_REG\_GZ só precisam de 8 bits, mas foram reservados 16 bits para facilitar a leitura e o mapa.

As temperaturas inicial e final permitem ver se houve grande variação de temperatura durante a calibração

Ajustes do Magnetômetro:

**ASAX, ASAY, ASAZ:** Sensitivity Adjustment values (ASA tem apenas 1 byte e parece não ter sinal). Exemplo para hx, idem para hy e hz.

**CALIBRAÇÃO**: Para ter um pouco de precisão, os valores de offset e ajuste estão multiplicados por 10. Então, primeiro é preciso dividi-los por 10 antes de usá-los.

**CALIBRAÇÃO AO LIGAR O CARRO**

Toda vez que o carro for ligado, é executada a chamada “Calibração ao ligar o carro”. Ela será guardada na SRAM e copiada para a Flash em caso de acidente. Na SRAM (ou Flash) ocupa o bloco de 416 bytes (0x3 7E60 🡪 0x3 7FFF). Este bloco é marcado com as constantes:

#define CXP\_ADR\_INI 0x37E60L //Início área de config da Caixa Preta (sobram 416 bytes)

#define CXP\_ADR\_FIM 0x38000L //Fim área de conficuração da Caixa Preta

Para simplificar os acessos, os dados gravados nesta área são de 16 bits, 32 bits ou strings terminadas em zero. Mesmo quando a informação é de apenas um byte, usam-se 2 bytes. As duas constantes abaixo padronizam em 16 bits as possibilidades de sim ou não.

#define COD\_SIM 0x5353 //2x ASCII(S) Afirmativo

#define COD\_NAO 0x4E4E //2x ASCII(N) Negativo

Tabela 1. Parâmetros do Setor de Configuração ao Ligar e Operação (SRAM e FLASH).

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Offset** | **Bytes** | **Tipo** | **Exemp** | **Nome** | **Descrição** |
| 0 | 2 | 16 bits | NN | OP\_OK | Fez calibração ao ligar? OK / NOK |
| 0 | 2 | 16 bits | NN | OP\_BATEU | (NN) 🡪 pronta, (SS) 🡪 acidentada |
| 2 | 2 | 16 bits | SS | OP\_ST\_OK | Self-Test: (SS) 🡪 OK, (NN) 🡪 NOK |
| 2 | 2 | 16 bits | SS | OP\_STH\_OK | Self-Test Mag: (SS) 🡪 OK, (NN) 🡪 NOK |
| 2 | 2 | 16 bits | SS | OP\_CF\_OK | Calib de Fabr: (SS) 🡪 OK, (NN) 🡪 NOK |
| 2 | 2 | 16 bits | SS | OP\_CFH\_OK | Mag fez Calib de Fabr: OK, NOK |
| - | - |  |  |  | **Acel e Giro - Calibra ao ligar** |
| 4 | 2 | 16 bits |  | OPC\_FREQ\_AG | Freq de amostragem (100,..., 1000 Hz) |
| 4 | 2 | 16 bits |  | OPC\_BW\_AG | Banda do filtro (5,10, ..., 260 Hz) |
| 6 | 2 | 16 bits |  | OPC\_ESC\_AC | Escala acelerômetro (1, 2, 4, 8 g) |
| 6 | 2 | 16 bits |  | OPC\_ESC\_GI | Escala giroscópio (250, ..., 2000) |
| 4 | 2 | 16 bits |  | OPC\_QTD\_AG | Qtd de medidas para a média |
| 8 | 2 | 16 bits |  | OPC\_AX | Média AX |
| 10 | 2 | 16 bits |  | OPC\_AY | Média AY |
| 12 | 2 | 16 bits |  | OPC\_AZ | Média AZ |
| 14 | 2 | 16 bits |  | OPC\_TP | Média TP |
| 16 | 2 | 16 bits |  | OPC\_GX | Média GX |
| 18 | 2 | 16 bits |  | OPC\_GY | Média GY |
| 20 | 2 | 16 bits |  | OPC\_GZ | Média GZ |
| - | - |  |  |  | **Magnet - Calibra ao ligar** |
| 22 | 2 | 16 bits |  | OPC\_QTD\_MG | Qtd de medidas para a média |
| 24 | 2 | 16 bits |  | OPC\_ESC\_MG | Escala do magnetômetro |
| 26 | 2 | 16 bits |  | OPC\_HX | Média HX |
| 28 | 2 | 16 bits |  | OPC\_HY | Média HY |
| 30 | 2 | 16 bits |  | OPC\_HZ | Média HZ |
| - | - |  |  |  | **Acel, Giro e Mag - Operação** |
| 32 | 2 | 16 bits |  | OP\_FREQ\_AG | Freq de amostragem (100,..., 1000 Hz) |
| 4 | 2 | 16 bits |  | OP\_BW\_AG | Banda do filtro (5,10, ..., 260 Hz) |
| 32 | 2 | 16 bits |  | OP\_ESC\_AC | Escala acelerômetro (1, 2, 4, 8 g) |
| 34 | 2 | 16 bits |  | OP\_ESC\_GI | Escala giroscópio (250, ..., 2000) |
| 36 | 2 | 16 bits |  | OP\_ESC\_MG | Escala Magnetômetro |
| 38 | 2 | 16 bits |  | OP\_LIM\_AX | Limiar disparo AX (valor absoluto) |
| 40 | 2 | 16 bits |  | OP\_LIM\_AY | Limiar disparo AY (valor absoluto) |
| 44 | 2 | 16 bits |  | OP\_LIM\_AZ | Limiar disparo AZ (valor absoluto) |
| 46 | 2 | 16 bits |  | OP\_LIM\_GX | Limiar disparo GX (valor absoluto) |
| 48 | 2 | 16 bits |  | OP\_LIM\_GY | Limiar disparo GY (valor absoluto) |
| 50 | 2 | 16 bits |  | OP\_LIM\_GZ | Limiar disparo GZ (valor absoluto) |
| - | - |  |  |  | **Quem Disparou ?** |
| 52 | 4 | 32 bits |  | OP\_MPU\_ADR | Endereço MPU na SRAM do disparo |
| 52 | 4 | 32 bits |  | OP\_GPS\_ADR | Endereço GPS na SRAM do disparo |
| 52 | 2 | 16 bits |  | OP\_DISP\_TP | Temperatura no instante do disparo |
| 52 | 2 | 16 bits | NN | OP\_DISP\_AX | AX Disparou (SS = sim, NN = não) |
| 54 | 2 | 16 bits | NN | OP\_DISP\_AY | AY Disparou (SS = sim, NN = não) |
| 56 | 2 | 16 bits | NN | OP\_DISP\_AZ | AZ Disparou (SS = sim, NN = não) |
| 58 | 2 | 16 bits | NN | OP\_DISP\_GX | GX Disparou (SS = sim, NN = não) |
| 60 | 2 | 16 bits | NN | OP\_DISP\_GY | GY Disparou (SS = sim, NN = não) |
| 62 | 2 | 16 bits | NN | OP\_DISP\_GZ | GZ Disparou (SS = sim, NN = não) |
| 62 | 2 | 16 bits | NN | OP\_BRK | Aquisição interrompida (SS,NN) |
| 62 | 4 | 32 bits | NN | OP\_ULT\_ADR | Último endereço gravado pelo MPU |
| - | - |  |  |  | **Data e Hora acidente (gps\_dados)** |
| 64 | 8 | String |  | OP\_AC\_DATA | Data do acidente: ddmmyy0 |
| 72 | 12 | String |  | OP\_AC\_HORA | Hora do acidente: hhmmss.sss0 |
| 84 | 73 | Char |  |  | 0xFF |
| ... | ... | Char |  |  | 0xFF |
| 415 | 415 | Char |  |  | 0xFF |

Sugestão para comunicação com a Caixa Preta (ainda por fazer)

|  |  |  |
| --- | --- | --- |
| **Cmdo.** | **Param.** | **Descrição** |
| ? | - | Enviar configuração. |
| I | - | Inverter estado acidentado  (T 🡪 F ou de F🡪T) |
| R | - | Zerar toda a Flash |
| L | - | Ler toda a memória |
| X | n | Limiar acelerômetro eixo X |
| Y | n | Limiar acelerômetro eixo Y |
| Z | n | Limiar acelerômetro eixo Z |
| x | n | Limiar giroscópio eixo X |
| y | n | Limiar giroscópio eixo Y |
| z | n | Limiar giroscópio eixo Z |
|  |  |  |

Funções

|  |  |  |
| --- | --- | --- |
| void | setup | (void) |
| void | loop | (void) |
| byte | sel\_modo | (char \*msg[], byte total) |
|  |  |  |
|  |  |  |

* void **setup** (void)
* void **loop** (void)
* byte **sel\_modo** (char \*msg[], byte total)

Retorna o número da opção selecionada. Recebe um vetor de ponteiros para as mensagens de cada opção e o total de opções. É usada para selecionar o modo de Operação e o tipo de Teste.

**Defs.h**

// TESTE

#define TESTE\_TOT 17 //Modos de teste: 1, 2 , ..., 17

#define TESTE\_0 0 //Opera

#define TESTE\_1 1 //LEDs

#define TESTE\_2 2 //LCD

#define TESTE\_3 3 //Teclado

#define TESTE\_4 4 //TWI

#define TESTE\_5 5 //Acel e giro

#define TESTE\_6 6 //Magnetometro

#define TESTE\_7 7 //SRAM

#define TESTE\_8 8 //FLAH

#define TESTE\_9 9 //GPS Tudo

#define TESTE\_10 10 //GPS RMC GSA

#define TESTE\_11 11 //GPS U-Center

#define TESTE\_12 12 //MPU-->Matlab

#define TESTE\_13 13 //BlueTooth

#define TESTE\_14 14 //Livre

#define TESTE\_15 15 //Livre

#define TESTE\_16 16 //Livre

#define TESTE\_17 17 //Livre

// OPERA

#define OPERA\_TOT 9 //Modos de teste: 1, 2 , ..., 9

#define OPERA\_0 0 //Teste

#define OPERA\_1 1 //Aqusição de Dados

#define OPERA\_2 2 //Livre

#define OPERA\_3 3 //Livre

#define OPERA\_4 4 //Livre

#define OPERA\_5 5 //Calibração de Fábrica

#define OPERA\_6 6 //Calibração do Magnetômetro

#define OPERA\_7 7 //Livre

#define OPERA\_8 8 //Livre

#define OPERA\_9 9 //Livre

**Globs.h**

// TESTE - Mensagens do modo de teste

char \*teste\_msg[]={ "0-Vai para Opera", //0

"1-LEDs", //1

"2-LCD", //2

"3-Teclado", //3

"4-TWI (I2C)", //4

"5-Acel e giro", //5

"6-Magnetometro", //6

"7-SRAM", //7

"8-FLASH", //8

"9-GPS: Tudo", //9

"10-GPS: Interpreta", //10

"11-GPS:U-Center", //11

"12-MPU-->MatLab", //12

"13-Blue Tooth", //13

"14-BT - Cmds AT", //14

"15-Vazio", //15

"16-Vazio", //16

"17-Vazio"}; //17

// OPERA - Mensagens do modo de Operação

char \*opera\_msg[]={ "0-Vai para Teste", //0

"1-Adquirir Dados", //1

"2-Vazio", //2

"3-vazio", //3

"4-Vazio", //4

"5-Calibra Fab", //5

"6-Calibra Mag", //6

"7-Vazio", //7

"8-Vazio", //8

"9-Vazio"}; //9

**CXP**

// Selecionar o modo

// Serve selecionar modo de Operação

// Serve selecionar modo de Teste

// A linha 0 é preparada por quem chama

// Usa as linhas 1, 2 e 3

byte sel\_modo(char \*msg[], byte total){

char prov=total; //provisório = 1, 2, ..., OPERA\_TOT

byte tecla,aux;

byte ser;

byte fase; //Indicar se estava recebendo nr pela serial

fase=0;

prov=0;

while(TRUE){

lcd\_apaga\_lin(1);

lcd\_apaga\_lin(2);

lcd\_apaga\_lin(3);

lcd\_str(2,0,"-->");

aux=prov;

lcd\_str(1,3,msg[aux++]);

if (aux>total) aux=0;

lcd\_str(2,3,msg[aux]);

ser\_str(msg[aux++]); ser\_crlf(1);

if (aux>total) aux=0;

lcd\_str(3,3,msg[aux]);

//Esperar tecla ou serial

while(TRUE){

if ( sw\_tira(&tecla) == TRUE) break;

if (seri\_letra(&ser) == TRUE) return ser;

if (seri\_num8(&ser) == TRUE) return ser;

}

switch(tecla){

case SW\_SUP: prov--; break;

case SW\_INF: prov++; break;

case SW\_SEL: if (++prov>total) prov=0;

return prov;

}

ser\_crlf(1);

if (prov>total) prov=0;

if (prov==-1) prov=total;

}

}

**Ideias sobre o projeto Caixa Preta**

1g = 9,80665 m/s2.

(INMETRO) Aceleração da gravida em Brasília (15°45'43.2"S 47°52'25.3"W) = 9,7808439 m/s2.

<https://www.researchgate.net/publication/282913685_Determinacao_Experimental_da_Aceleracao_da_Gravidade_com_Pendulo_Simples_e_Cameras_Digitais>

Em Brasília estamos com 9,7808439/9,80665 = 99,7369%.

Com MPU na escala de +/- 2g, deveríamos ler: (32.768/2)\*0,997369 = 16.341 = 0x3FD5.