**Biblioteca GPS para a Caixa Preta**

Versão 1.0, 13/04/2020

Funções

|  |  |  |
| --- | --- | --- |
|  |  |  |
| void | gps\_extrai | (byte \*vt) |
| byte | gps\_come\_vg | (byte ini, byte \*vt, byte qtd) |
| void | gps\_cpy\_vg | (byte \*fonte, byte \*dest) |
| byte | gps\_idtf | (byte \*vt) |
| void | gps\_str | (byte \*msg) |
| void | gps\_char | (byte dt) |
| void | gps\_config | (long br) |
|  | ISR(USART3\_TX\_vect) |  |
|  | ISR(USART3\_RX\_vect) |  |
|  |  |  |
|  |  |  |

* void **gps\_extrai** (byte \*vt)

Extrair os das de uma mensagem do GPS.

Identifica o tipo de mensagem e extrai os dados de interesse.

Atualiza os dados no vetor **gps\_dados.**

* byte **gps\_come\_vg** (byte ini, byte \*vt, byte qtd)

Avançar uma certa quantidade (**qtd**) de vírgulas a partir da posição **ini** no vetor **vt**.

Retorna indexador para a primeira posição após a vírgula.

* void **gps\_cpy\_vg** (byte \*fonte, byte \*dest)

Copiar trecho da **string** fonte para a string **dest**.

Para (interrompe) quando encontra uma vírgula ou "\*". Coloca um zero no final de **dest**.

* byte **gps\_idtf** (byte \*vt)

Identifica o tipo de mensagens que está no vetor vt.

Retorna número correspondente:

0=GPS\_NADA 1=GPS\_RMC 2=GPS\_VTG 3=GPS\_GGA 4=GPS\_GSA 5=GPS\_GSV 6=GPS\_GLL

* void **gps\_str** (byte \*msg)

Enviar msg para GPS.

* void **gps\_char** (byte dt)

Enviar um char para o GPS. Não usa interrupção.

* void **gps\_config** (long br)

Configurar porta serial 3. Não habilita TX e nem RX. Não habilita interrupções.

* ISR(**USART3\_TX\_vect**)

TX3: Interrupção por dado enviado

* ISR(**USART3\_RX\_vect**)

RX3: Interrupção por dado recebido.

Se **gps\_msg\_fase** = 0 🡪 armazena em **gps\_msg\_0** [gps\_msg\_ix++].

Se **gps\_msg\_fase** = 1 🡪 armazena em **gps\_msg\_1** [gps\_msg\_ix++].

Quando recebe uma mensagem completar (termina com 0xD) marca o final com ‘\0’ (ver posição exata) e faz **gps\_msg\_ok=TRUE**.

------------------------------- Sobre o GPS ----------------------------------

A recepção do GPS usa 2 vetores alternadamente, enquanto um é preenchido, o outro é analisado, depois troca.

byte **gps\_msg\_0**[GPS\_MSG\_TAM], **gps\_msg\_1**[GPS\_MSG\_TAM];

byte **gps\_msg\_fase** 🡪 indicar qual vetor recebe dados que chegam pela porta serial RX3

byte **gps\_msg\_ok** 🡪 indicar que completou a recepção de uma mensagem GPS

byte **gps\_msg\_ix** 🡪 indexador para escrever nos vetores

|  |  |  |
| --- | --- | --- |
| **gps\_msg\_flase** | **gps\_msg\_0** | **gps\_msg\_1** |
| FALSE | Recebendo | Analisando |
| TRUE | Analisando | Recebendo |

Mensagens vindas do GPS têm o formato abaixo

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| $ | G | P | ... | ... |  |  | xx | CR | LF |
| 24 | 47 | 50 | ... | .. |  |  | xx | 0D | 0A |

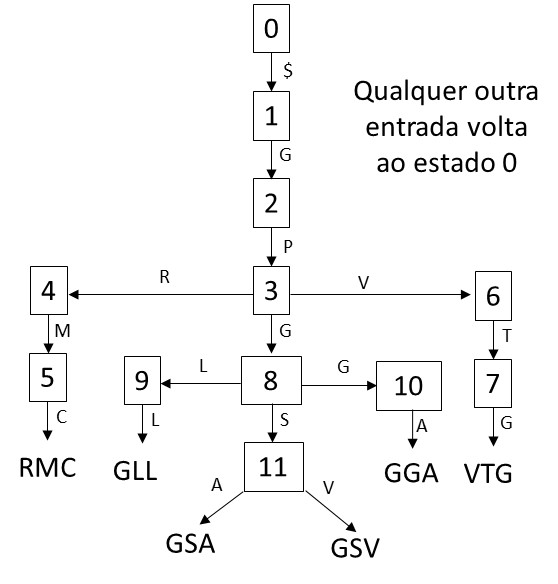
Iniciam com $ e terminam com 0xD e0xA.

Após o armazenamento no vetor (**gps\_msg\_0** ou **gps\_msg\_1**), é colocado um ‘\0’ no lugar do 0xD final.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| $ | G | P | ... | ... |  |  | xx | ‘\0’ | LF |
| 24 | 47 | 50 | ... | .. |  |  | xx | ‘\0’ | 0A |

Iniciam com $ e terminam com 0x00.

As 6 letras iniciais servem para identificar o tipo de mensagem (GPRMC, GPGSA, ...). O tipo de mensagem é descoberto usando o diagrama de estados abaixo.



Resumo das mensagens de onde se retiraram informações

-------------------------------------------------------- GPRMC --------------------------------------------------------

$GPRMC,hhmmss,status,latitude,N,longitude,E,spd,cog,ddmmyy,mv,mvE,mode\*cs<CR><LF>

$GPRMC,083559.00,A,4717.11437,N,00833.91522,E,0.004,77.52,091202,,,A\*57

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 |
| $GPRMC, | hhmmss.sss, | Stat, | Lat:ddmm.mmmmm, | N/S, | Long:dddmm.mmmmm, | E/W, |
| $GPRMC, | 083559.00, | A, | 4717.11437, | N, | 00833.91522, | E, |
| 7 | 12 | 2 | 11 | 2 | 12 | 2 |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 |
| Speed:ddd.ddd, | Curso:ddd.ddd, | Data:ddmmyy | Mv, | mvE, | Modo | \*Check | CR LF |
| 0.004, | 77.52, | 091202, | , | , | A | \*57 | 0xD 0xA |
| ?8 | ?8 | 7 | ?8 | ?2 | 1 | 3 | 2 |

Tamanho = 80 bytes, vou usar tamanho 100.

Lido com Arduino: $GPRMC,131732.00,A,1548.62581,S,04748.65809,W,0.299,,260120,,,A\*72

--------------------------------------------------------- GPGSA --------------------------------------------------------

$GPGSA,Smode,FS{,sv},PDOP,HDOP,VDOP\*cs<CR><LF>

$GPGSA,A,3,23,29,07,08,09,18,26,28,,,,,1.94,1.18,1.54\*0D

Mostra identificador de até 12 satélites

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| $GPGSA, | Smode, | Fix, | Sat1, | Sat2, | Sat3, | Sat4, | Sat5, | Sat6, | Sat7, | Sat8 |
| $GPGSA, | A, | 3, | 23, | 29, | 07, | 08, | 09, | 18, | 26, | 28, |
| 7 | 2 | 2 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 |
| Sat9, | Sat10 | Sat11, | Sat12, | PDOP, | HDOP, | VDOP | \*Check | CR LF |
| 18, | 18, | 18, | 18, | 1.94, | 1.18, | 1.54 | \*0D | 0xD 0xA |
| 3 | 3 | 3 | 3 | 5 | 5 | 5 | 3 | 2 |

Tamanho = 67 bytes, vou usar tamanho 100.

Lido com Arduino: $GPGSA,A,3,07,09,16,23,04,01,,,,,,,8.16,1.24,8.06\*09

-------------------------------------------------------- GPVTG --------------------------------------------------------

$GPVTG,cogt,T,cogm,M,sog,N,kph,K,mode\*cs<CR><LF>

$GPVTG,77.52,T,,M,0.004,N,0.008,K,A\*06

Mostra curso e velocidade

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 |
| $GPVTG, | Curso T  Cogt:ddd.dd, | Fix, | Curso M  Cogm:ddd.dd, | Fix | Veloc Nós  Sog:ddd.ddd, | Unidade |
| $GPVTG, | 77.52, | T, | ddd.dd, | M, | 0.004, | N, |
| 7 | 7 | 2 | 7 | 3 | 8 | 2 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 7 | 8 | 9 | 10 | 11 |
| Veloc kph  Sog:ddd.ddd, | Unidade | Modo | \*Check | CR LF |
| 0.008 | K, | A, | \*06 | 0xD 0xA |
| 8 | 2 | 2 | 3 | 2 |

Tamanho = bytes, vou usar tamanho 100.

Lido com Arduino: $GPVTG,,T,,M,0.079,N,0.146,K,A\*2E

-------------------------------------------------------- GPGGA --------------------------------------------------------

$GPGGA,hhmmss.ss,Latitude,N,Longitude,E,FS,NoSV,HDOP,msl,m,Altref,m,DiffAge,DiffStation\*cs<CR><LF>

$GPGGA,092725.00,4717.11399,N,00833.91590,E,1,8,1.01,499.6,M,48.0,M,,0\*5B

Mostra ...

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| $GPGGA, | hhmmss.sss, | Lat:ddmm.mmmmm, | N/S, | Long:dddmm.mmmmm, | E/W, | FS, | Nr sat:dd, |
| $GPGGA, | 083559.00, | 4717.11437, | N, | 00833.91522, | E, | 1, | 8, |
| 7 | 12 | 11 | 2 | 12 | 2 | 2 |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 8 | 9 | 10 | 11 | 12 |
| HDOP, | Alt Msl:dddd.ddd, | uMsl | Altref:dddd.ddd, | Usep |
| 1.18, | 1.01, | M, | 48.0, | M |
| 5 | 9 | 2 | 9 | 2 |

|  |  |  |  |
| --- | --- | --- | --- |
| 13 | 14 | 15 | 16 |
| Diffage, | DiffStation | \*Check | CR LF |
| S, | 0 | \*57 | 0xD 0xA |
| 2 | 2 | 3 | 2 |

Tamanho = bytes, vou usar tamanho 100.

Lido com Arduino: $GPGGA,185326.00,1548.63054,S,04748.65655,W,1,06,2.06,1052.0,M,-11.8,M,,\*4F

--------------------------------------- Vetor usado para separar o que é importante ------------------------------

Vetor **gps\_dados[GPS\_DADOS\_TAM]** armazena as informações extraídas do GPS

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| RMC | RMC | RMC | RMC | RMC | RMC | RMC |
| 0 | 2 | 13 | 20 | 31 | 33 | 45 |
| Status | Hora | Data | Latitude | N/S | Longitude | E/W |
| A0 | hhmmss.sss0 | ddmmyy0 | ddmm.mmmmm0 | N0 | dddmm.mmmmm0 | E0 |
| A | 083559.00 | 091202 | 4717.11437 | N | 00833.91522 | E |
| 2 | 11 | 7 | 11 | 2 | 12 | 2 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| RMC | RMC | GSA | GSA | GSA | VTG | VTG |
| 47 | 55 | 63 | 69 | 75 | 81 | 88 |
| Velocidade Nós | Curso | PDOP | HDOP | VDOP | Speed km/h | Unidade |
| ddd.ddd0 | ddd.ddd0 | dd.dd0 | dd.dd0 | dd.dd0 | xxx.xx0 | K0 |
| 0.004 | 77.52 | 99.99 | 99.99 | 99.99 | ?125.12 | K |
| 8 | 8 | 6 | 6 | 6 | 7 | 2 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| VTG | GGA | GGA | GGA | - |  |  |
| 90 | 92 | 95 | 102 | 105 | 110 |  |
| ?Fix? | Qtd Sat | Altitude | Unidade | Adr SRAM |  |  |
| d0 | dd0 | dddd.d0 | m0 | HHHHH0 |  |  |
| 1 | 4 | 627.4 | m | 3F4CA |  |  |
| 2 | 3 | 7 | 2 | 5 |  |  |

**Adr SRAM** indica onde estava o ponteiro de gravação da SRAM quando essa mensagem do GPS foi gravada. Vai permitir a sincronização do GPS com o MPU.

// Marcar posição de cada um dos parâmetros guardados em gps\_dados[GPS\_DADOS\_TAM]

#define GPS\_STATUS 0 //2 bytes

#define GPS\_HORA (GPS\_STATUS+2) //11 bytes

#define GPS\_DATA (GPS\_HORA+11) //7 bytes

#define GPS\_LAT (GPS\_DATA+7) //11 bytes

#define GPS\_NS (GPS\_LAT+11) //2 bytes

#define GPS\_LONG (GPS\_NS+2) //12 bytes

#define GPS\_EW (GPS\_LONG+12) //2 bytes

#define GPS\_VEL\_NOS (GPS\_EW+2) //8 bytes

#define GPS\_CURSO (GPS\_VEL\_NOS+8) //8 bytes

#define GPS\_PDOP (GPS\_CURSO+8) //6 bytes

#define GPS\_HDOP (GPS\_PDOP+6) //6 bytes

#define GPS\_VDOP (GPS\_HDOP+6) //6 bytes

#define GPS\_VEL\_KPH (GPS\_VDOP+6) //7 bytes

#define GPS\_VEL\_UN (GPS\_VEL\_KPH+7) //2 bytes

#define GPS\_FIX (GPS\_VEL\_UN+2) //2 bytes

#define GPS\_QTD\_SAT (GPS\_FIX+2) //3 bytes sem uso

#define GPS\_ALT (GPS\_QTD\_SAT+3) //7 bytes

#define GPS\_ALT\_UN (GPS\_ALT+7) //2 bytes

#define GPS\_ADR\_SRAM (GPS\_ALT\_UN+2) //5 bytes

--------------------------------------------------------------------------------------------------

<https://www.academia.edu/38817589/GYNEO6MV2_GPS_Module_with_Arduino>

Decodificador on line <https://rl.se/gprmc>

<https://portal.u-blox.com/s/question/0D52p00008HKDUlCAP/disable-nmea-sentences-on-uart>

<https://portal.u-blox.com/s/question/0D52p00008HKCN4/disable-nmea-sentence>

<https://forum.arduino.cc/index.php?topic=232896.0>

Comandos: <https://ukhas.org.uk/guides:falcom_fsa03>

**U-Center inicia com as mensagens**

B5 62 0A 00 34 B5 62 0A 04 34

B5 62 0A 04 34 B5 62 0A 00 34

B5 62 0A 04 34 B5 62 0A 04 34

o nível dos DOPs, a tabela a seguir mostra as faixas e níveis de precisão.

|  |  |  |
| --- | --- | --- |
| **Nível DOP** | **Qualidade** | **Descrição** |
| < 1 | Ideal | Nível de confiança mais alto; máxima precisão possível em todos os momentos. |
| 1-2 | Excelente | Medições precisas |
| 2-5 | Bom | Medições com precisão adequadas |
| 5-10 | Moderado | Qualidade moderada. Correção recomendada |
| 10-20 | Fraco | Nível de confiança baixo. Considere descartar dados |
| >20 | Ruim | Precisão muito baixa. Erros podem atingir 300 metros |