Biblioteca acesso ao GPS

<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

Mensagens do GPS terminam com [0xD 0xA] ou [‘\r’ ‘\n’]

Funções

|  |  |  |
| --- | --- | --- |
| byte | gps\_rmc\_ser | (char \*vt) |
| byte | gps\_gsa\_ser | (char \*vt) |
| void | gps\_str | (byte \*msg) |
| void | gps\_char | (byte dt) |
| void | gps\_rx\_fila\_config | (void) |
| char | gps\_rx\_poe | (char cha) |
| void | gps\_fila\_rx\_cheia | (void) |
| char | gps\_rx\_tira | (char \*cha) |
| void | gps\_config | (long br) |
|  | ISR(USART3\_TX\_vect) |  |
|  | ISR(USART3\_RX\_vect) |  |
|  |  |  |

* byte **gps\_rmc\_ser** (char \*vt)

GP RMC - Recebe uma string e se for RMC, a imprime.

Retorna TRUE se era RMC (imprime).

Retorna FALSE se não era (nada imprime).

Seq Vazia = $GPRMC,,V,,,,,,,,,,N\*53

01234567890123456789012

* byte **gps\_gsa\_ser** (char \*vt)

GP GSA - Recbe uma string e se for GSA, a imprime.

Retorna TRUE se era GSA (imprime).

Retorna FALSE se não era (nada imprime).

Seq Vazia = $GPGSA,A,3,,,,,,,,,,,,N\*53

01234567890123456789012

* 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\_rx\_fila\_config** (void)

Inicializar fila GPS RX serial.

* char **gps\_rx\_poe** (char cha)

Colocar um byte na fila RX do GPS.

* void **gps\_fila\_rx\_cheia** (void)

Mensagem de Fila RX Cheia do GPS.

* char **gps\_rx\_tira** (char \*cha)

Tirar um byte da fila RX do GPS.

* 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

volatile char gps\_tx\_fila[GPS\_TX\_FILA\_TAM]; //Espaço para a fila serial de TX

volatile byte gps\_tx\_pin, gps\_tx\_pout; //Ponteiros para usar a fila

volatile byte gps\_tx\_ok; //Indica que terminou transmissão

volatile char gps\_rx\_fila[GPS\_RX\_FILA\_TAM]; //Espaço para a fila serial de RX

volatile byte gps\_rx\_pin, gps\_rx\_pout; //Ponteiros para usar a fila

volatile byte gps\_rx\_ok; //Indica que terminou recepção

volatile byte gps\_msg\_0[200]; //(fase=0) Buffer 0 usado pela interrupção para receber GPRMC

volatile byte gps\_msg\_1[200]; //(fase=1) Buffer 1 usado pela interrupção para receber GPRMC

volatile byte gps\_msg\_ix; //Indexador para escrever nos buffers;

volatile byte gps\_msg\_fase; //Fase para receber GPRMC (0=gps\_rmc\_0 e 1=gps\_rmc\_1);

volatile byte gps\_msg\_ok; //Indica que completou o recebimento de uma mensagem;

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

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, | hhmmmss.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 |

Tamanh0 = 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

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| $GPGSA, | Smode, | Fix, | Sat1, | Sat2, | Sat3, | Sat4, | Sat5, | Sat6, | Sat7, | Sat8 |
| $GPRMC, | 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