Arduino Naza Decoder Driver

Generated by Doxygen 1.8.9.1

Tue Dec 29 2015 15:12:54

Contents

1 Class Index

1.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

2 File Index

2.1 File List

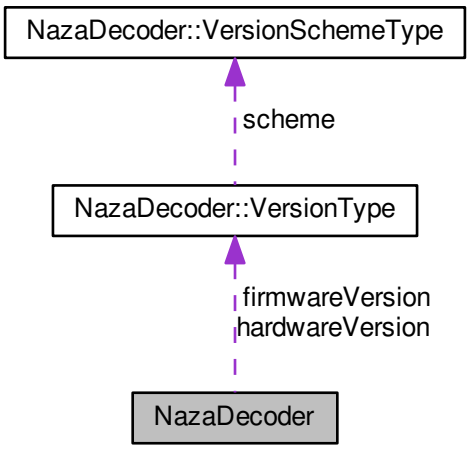
Here is a list of all files with brief descriptions:

3 Class Documentation

3.1 NazaDecoder Class Reference

`#include <NazaDecoder.h>`

Collaboration diagram for NazaDecoder:



Classes

- struct VersionSchemeType
- union VersionType

Public Types

- enum GPSPayloadPosition {
  NAZA_MESSAGE_POS_DT = 0x04 - MESSAGE_HEADER_SIZE, NAZA_MESSAGE_POS_LO = 0x08 - MESSAGE_HEADER_SIZE, NAZA_MESSAGE_POS_LA = 0x0c - MESSAGE_HEADER_SIZE, NAZA_M↩ ESSAGE_POS_AL = 0x10 - MESSAGE_HEADER_SIZE,
  NAZA_MESSAGE_POS_HA = 0x14 - MESSAGE_HEADER_SIZE, NAZA_MESSAGE_POS_VA = 0x18 - MESSAGE_HEADER_SIZE, NAZA_MESSAGE_POS_NV = 0x20 - MESSAGE_HEADER_SIZE, NAZA_↩ MESSAGE_POS_EV = 0x24 - MESSAGE_HEADER_SIZE,
  NAZA_MESSAGE_POS_DV = 0x28 - MESSAGE_HEADER_SIZE, NAZA_MESSAGE_POS_PD = 0x2c - MESSAGE_HEADER_SIZE, NAZA_MESSAGE_POS_VD = 0x2e - MESSAGE_HEADER_SIZE, NAZA_↩ MESSAGE_POS_ND = 0x30 - MESSAGE_HEADER_SIZE,
  NAZA_MESSAGE_POS_ED = 0x32 - MESSAGE_HEADER_SIZE, NAZA_MESSAGE_POS_NS = 0x34 -

MESSAGE_HEADER_SIZE, NAZA_MESSAGE_POS_FT = 0x36 - MESSAGE_HEADER_SIZE, NAZA_M↩ESSAGE_POS_SF = 0x38 - MESSAGE_HEADER_SIZE,
NAZA_MESSAGE_POS_XM = 0x3b - MESSAGE_HEADER_SIZE, NAZA_MESSAGE_POS_SN = 0x3c - MESSAGE_HEADER_SIZE, NAZA_MESSAGE_POS_CS = 0x3e - MESSAGE_HEADER_SIZE }

- enum CompassPayloadPosition { NAZA_MESSAGE_POS_CX = 0x04 - MESSAGE_HEADER_SIZE, NAZ↩A_MESSAGE_POS_CY = 0x06 - MESSAGE_HEADER_SIZE, NAZA_MESSAGE_POS_CZ = 0x08 - ME↩SSAGE_HEADER_SIZE }
- enum ModuleVersionPayloadPosition { NAZA_MESSAGE_POS_FW = 0x08 - MESSAGE_HEADER_SIZE, NAZA_MESSAGE_POS_HW = 0x0c - MESSAGE_HEADER_SIZE }
- enum MessageType { NAZA_MESSAGE_NONE_TYPE = 0x00, NAZA_MESSAGE_GPS_TYPE = 0x10, N↩AZA_MESSAGE_COMPASS_TYPE = 0x20, NAZA_MESSAGE_MODULE_VERSION_TYPE = 0x30 }
- enum MessageSize { NAZA_MESSAGE_GPS_SIZE = 0x3a, NAZA_MESSAGE_COMPASS_SIZE = 0x06, NAZA_MESSAGE_MODULE_VERSION_SIZE = 0x0c }
- enum FixType { NO_FIX = 0, FIX_2D = 2, FIX_3D = 3, FIX_DGPS = 4 }

Public Member Functions

- NazaDecoder ()
- uint8_t decode (int16_t input)
- double getLat ()
- double getLon ()
- double getGpsAlt ()
- double getSpeed ()
- FixType getFixType ()
- uint8_t getNumSat ()
- double getHeading ()
- double getCog ()
- double getGpsVsi ()
- double getHdop ()
- double getVdop ()
- uint8_t getYear ()
- uint8_t getMonth ()
- uint8_t getDay ()
- uint8_t getHour ()
- uint8_t getMinute ()
- uint8_t getSecond ()
- VersionType getFirmwareVersion ()
- VersionType getHardwareVersion ()
- uint8_t isLocked ()

Private Member Functions

- int32_t pack4 (uint8_t i, uint8_t mask)
- int16_t pack2 (uint8_t i, uint8_t mask)
- void updateChecksum (int16_t input)

Private Attributes

- int16_t payload [58]
- int16_t seq
- int16_t cnt
- int16_t msgId
- int16_t msgLen
- uint8_t cs1

- uint8_t cs2
- int16_t magXMin
- int16_t magXMax
- int16_t magYMin
- int16_t magYMax
- double lon
- double lat
- double gpsAlt
- double spd
- FixType fix
- uint8_t sat
- double heading
- double cog
- double gpsVsi
- double hdop
- double vdop
- uint8_t year
- uint8_t month
- uint8_t day
- uint8_t hour
- uint8_t minute
- uint8_t second
- VersionType firmwareVersion
- VersionType hardwareVersion
- uint16_t lastLock
- uint8_t locked

3.1.1 Detailed Description

Definition at line 14 of file NazaDecoder.h.

3.1.2 Member Enumeration Documentation

3.1.2.1 enum NazaDecoder::CompassPayloadPosition

Enumerator

NAZA_MESSAGE_POS_CX

NAZA_MESSAGE_POS_CY

NAZA_MESSAGE_POS_CZ

Definition at line 78 of file NazaDecoder.h.

3.1.2.2 enum NazaDecoder::FixType

Enumerator

NO_FIX

FIX_2D

FIX_3D

FIX_DGPS

Definition at line 112 of file NazaDecoder.h.

3.1.2.3 enum NazaDecoder::GPSPayloadPosition

Enumerator

NAZA_MESSAGE_POS_DT
NAZA_MESSAGE_POS_LO
NAZA_MESSAGE_POS_LA
NAZA_MESSAGE_POS_AL
NAZA_MESSAGE_POS_HA
NAZA_MESSAGE_POS_VA
NAZA_MESSAGE_POS_NV
NAZA_MESSAGE_POS_EV
NAZA_MESSAGE_POS_DV
NAZA_MESSAGE_POS_PD
NAZA_MESSAGE_POS_VD
NAZA_MESSAGE_POS_ND
NAZA_MESSAGE_POS_ED
NAZA_MESSAGE_POS_NS
NAZA_MESSAGE_POS_FT
NAZA_MESSAGE_POS_SF
NAZA_MESSAGE_POS_XM
NAZA_MESSAGE_POS_SN
NAZA_MESSAGE_POS_CS

Definition at line 18 of file NazaDecoder.h.

3.1.2.4 enum NazaDecoder::MessageSize

Enumerator

NAZA_MESSAGE_GPS_SIZE
NAZA_MESSAGE_COMPASS_SIZE
NAZA_MESSAGE_MODULE_VERSION_SIZE

Definition at line 106 of file NazaDecoder.h.

3.1.2.5 enum NazaDecoder::MessageType

Enumerator

NAZA_MESSAGE_NONE_TYPE
NAZA_MESSAGE_GPS_TYPE
NAZA_MESSAGE_COMPASS_TYPE
NAZA_MESSAGE_MODULE_VERSION_TYPE

Definition at line 99 of file NazaDecoder.h.

3.1.2.6 enum NazaDecoder::ModuleVersionPayloadPosition

Enumerator

NAZA_MESSAGE_POS_FW
NAZA_MESSAGE_POS_HW

Definition at line 90 of file NazaDecoder.h.

3.1.3 Constructor & Destructor Documentation

3.1.3.1 NazaDecoder::NazaDecoder ( )

Definition at line 4 of file NazaDecoder.cpp.

3.1.4 Member Function Documentation

3.1.4.1 uint8_t NazaDecoder::decode (int16_t *input*)

Definition at line 92 of file NazaDecoder.cpp.

3.1.4.2 double NazaDecoder::getCog ( )

Definition at line 40 of file NazaDecoder.cpp.

3.1.4.3 uint8_t NazaDecoder::getDay ( )

Definition at line 64 of file NazaDecoder.cpp.

3.1.4.4 NazaDecoder::VersionType NazaDecoder::getFirmwareVersion ( )

Definition at line 80 of file NazaDecoder.cpp.

3.1.4.5 NazaDecoder::FixType NazaDecoder::getFixType ( )

Definition at line 28 of file NazaDecoder.cpp.

3.1.4.6 double NazaDecoder::getGpsAlt ( )

Definition at line 20 of file NazaDecoder.cpp.

3.1.4.7 double NazaDecoder::getGpsVsi ( )

Definition at line 44 of file NazaDecoder.cpp.

3.1.4.8 NazaDecoder::VersionType NazaDecoder::getHardwareVersion ( )

Definition at line 84 of file NazaDecoder.cpp.

3.1.4.9 double NazaDecoder::getHdop ( )

Definition at line 48 of file NazaDecoder.cpp.

3.1.4.10 double NazaDecoder::getHeading ( )

Definition at line 36 of file NazaDecoder.cpp.

3.1.4.11 uint8_t NazaDecoder::getHour ( )

Definition at line 68 of file NazaDecoder.cpp.

3.1.4.12 double NazaDecoder::getLat ( )

Definition at line 12 of file NazaDecoder.cpp.

3.1.4.13 double NazaDecoder::getLon ( )

Definition at line 16 of file NazaDecoder.cpp.

3.1.4.14 uint8_t NazaDecoder::getMinute ( )

Definition at line 72 of file NazaDecoder.cpp.

3.1.4.15 uint8_t NazaDecoder::getMonth ( )

Definition at line 60 of file NazaDecoder.cpp.

3.1.4.16 uint8_t NazaDecoder::getNumSat ( )

Definition at line 32 of file NazaDecoder.cpp.

3.1.4.17 uint8_t NazaDecoder::getSecond ( )

Definition at line 76 of file NazaDecoder.cpp.

3.1.4.18 double NazaDecoder::getSpeed ( )

Definition at line 24 of file NazaDecoder.cpp.

3.1.4.19 double NazaDecoder::getVdop ( )

Definition at line 52 of file NazaDecoder.cpp.

3.1.4.20 uint8_t NazaDecoder::getYear ( )

Definition at line 56 of file NazaDecoder.cpp.

3.1.4.21 uint8_t NazaDecoder::isLocked ( )

Definition at line 88 of file NazaDecoder.cpp.

3.1.4.22 int16_t NazaDecoder::pack2 (uint8_t *i*, uint8_t *mask*) [private]

Definition at line 240 of file NazaDecoder.cpp.

3.1.4.23 int32_t NazaDecoder::pack4 (uint8_t *i*, uint8_t *mask*) [private]

Definition at line 230 of file NazaDecoder.cpp.

3.1.4.24 void NazaDecoder::updateChecksum (int16_t *input*) [private]

Definition at line 7 of file NazaDecoder.cpp.

3.1.5 Member Data Documentation

3.1.5.1 int16_t NazaDecoder::cnt [private]

Definition at line 162 of file NazaDecoder.h.

3.1.5.2 double NazaDecoder::cog [private]

Definition at line 198 of file NazaDecoder.h.

3.1.5.3 uint8_t NazaDecoder::cs1 [private]

Definition at line 167 of file NazaDecoder.h.

3.1.5.4 uint8_t NazaDecoder::cs2 [private]

Definition at line 170 of file NazaDecoder.h.

3.1.5.5 uint8_t NazaDecoder::day [private]

Definition at line 210 of file NazaDecoder.h.

3.1.5.6 VersionType NazaDecoder::firmwareVersion [private]

Definition at line 215 of file NazaDecoder.h.

3.1.5.7 FixType NazaDecoder::fix [private]

Definition at line 189 of file NazaDecoder.h.

3.1.5.8 double NazaDecoder::gpsAlt [private]

Definition at line 183 of file NazaDecoder.h.

3.1.5.9 double NazaDecoder::gpsVsi [private]

Definition at line 201 of file NazaDecoder.h.

3.1.5.10 VersionType NazaDecoder::hardwareVersion [private]

Definition at line 216 of file NazaDecoder.h.

3.1.5.11 double NazaDecoder::hdop [private]

Definition at line 204 of file NazaDecoder.h.

3.1.5.12 double NazaDecoder::heading [private]

Definition at line 195 of file NazaDecoder.h.

3.1.5.13 uint8_t NazaDecoder::hour [private]

Definition at line 211 of file NazaDecoder.h.

3.1.5.14 uint16_t NazaDecoder::lastLock [private]

Definition at line 218 of file NazaDecoder.h.

3.1.5.15 double NazaDecoder::lat [private]

Definition at line 180 of file NazaDecoder.h.

3.1.5.16 uint8_t NazaDecoder::locked [private]

Definition at line 219 of file NazaDecoder.h.

3.1.5.17 double NazaDecoder::lon [private]

Definition at line 177 of file NazaDecoder.h.

3.1.5.18 int16_t NazaDecoder::magXMax [private]

Definition at line 172 of file NazaDecoder.h.

3.1.5.19 int16_t NazaDecoder::magXMin [private]

Definition at line 171 of file NazaDecoder.h.

3.1.5.20 int16_t NazaDecoder::magYMax [private]

Definition at line 174 of file NazaDecoder.h.

3.1.5.21 int16_t NazaDecoder::magYMin [private]

Definition at line 173 of file NazaDecoder.h.

3.1.5.22 uint8_t NazaDecoder::minute [private]

Definition at line 212 of file NazaDecoder.h.

3.1.5.23 uint8_t NazaDecoder::month [private]

Definition at line 209 of file NazaDecoder.h.

3.1.5.24 int16_t NazaDecoder::msgId [private]

Definition at line 163 of file NazaDecoder.h.

3.1.5.25 int16_t NazaDecoder::msgLen [private]

Definition at line 164 of file NazaDecoder.h.

3.1.5.26 int16_t NazaDecoder::payload[58] [private]

Definition at line 160 of file NazaDecoder.h.

3.1.5.27 uint8_t NazaDecoder::sat [private]

Definition at line 192 of file NazaDecoder.h.

3.1.5.28 uint8_t NazaDecoder::second [private]

Definition at line 213 of file NazaDecoder.h.

3.1.5.29 int16_t NazaDecoder::seq [private]

Definition at line 161 of file NazaDecoder.h.

3.1.5.30 double NazaDecoder::spd [private]

Definition at line 186 of file NazaDecoder.h.

3.1.5.31 double NazaDecoder::vdop [private]

Definition at line 207 of file NazaDecoder.h.

3.1.5.32 uint8_t NazaDecoder::year [private]

Definition at line 208 of file NazaDecoder.h.

The documentation for this class was generated from the following files:

- NazaDecoder.h
- NazaDecoder.cpp

3.2 NazaDecoder::VersionSchemeType Struct Reference

```
#include <NazaDecoder.h>
```

Public Attributes

- uint8_t revision
- uint8_t build
- uint8_t minor
- uint8_t major

3.2.1 Detailed Description

Definition at line 119 of file NazaDecoder.h.

3.2.2 Member Data Documentation

3.2.2.1 uint8_t NazaDecoder::VersionSchemeType::build

Definition at line 121 of file NazaDecoder.h.

3.2.2.2 uint8_t NazaDecoder::VersionSchemeType::major

Definition at line 123 of file NazaDecoder.h.

3.2.2.3 uint8_t NazaDecoder::VersionSchemeType::minor

Definition at line 122 of file NazaDecoder.h.

3.2.2.4 uint8_t NazaDecoder::VersionSchemeType::revision

Definition at line 120 of file NazaDecoder.h.

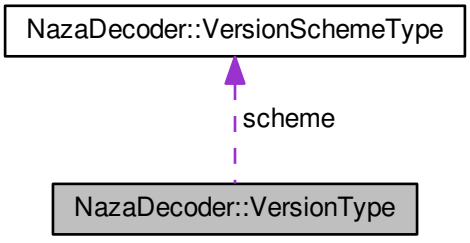
The documentation for this struct was generated from the following file:

- NazaDecoder.h

3.3 NazaDecoder::VersionType Union Reference

`#include <NazaDecoder.h>`

Collaboration diagram for NazaDecoder::VersionType:



Public Attributes

- uint32_t version

- VersionSchemeType scheme

3.3.1 Detailed Description

Definition at line 126 of file NazaDecoder.h.

3.3.2 Member Data Documentation

3.3.2.1 VersionSchemeType NazaDecoder::VersionType::scheme

Definition at line 128 of file NazaDecoder.h.

3.3.2.2 uint32_t NazaDecoder::VersionType::version

Definition at line 127 of file NazaDecoder.h.

The documentation for this union was generated from the following file:

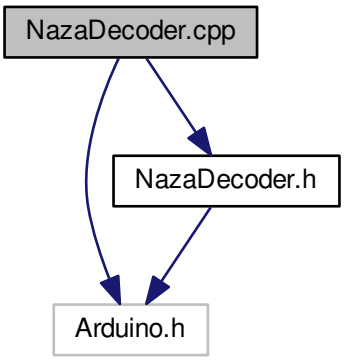
- NazaDecoder.h

4 File Documentation

4.1 NazaDecoder.cpp File Reference

```
#include <Arduino.h>
#include "NazaDecoder.h"
```

Include dependency graph for NazaDecoder.cpp:



4.2 NazaDecoder.cpp

```
00001 #include <Arduino.h>
00002 #include "NazaDecoder.h"
00003
00004 NazaDecoder::NazaDecoder()
00005         : seq(0), cnt(0), msgId(0), msgLen(0), cs1(0), cs2(0), magXMin(0), magXMax(0), magYMin(0), magYMax(
      0), lon(0), lat(0), gpsAlt(0), spd(0), fix(NO_FIX), sat(0), heading(0), cog(0), gpsVsi(0), hdop(0), vdop(0),
       year(0), month(0), day(0), hour(0), minute(0), second(0) {
00006 }
```

```
00007 void NazaDecoder::updateChecksum(int16_t input) {
00008     cs1 += input;
00009     cs2 += cs1;
00010 }
00011
00012 double NazaDecoder::getLat() {
00013     return lat;
00014 }
00015
00016 double NazaDecoder::getLon() {
00017     return lon;
00018 }
00019
00020 double NazaDecoder::getGpsAlt() {
00021     return gpsAlt;
00022 }
00023
00024 double NazaDecoder::getSpeed() {
00025     return spd;
00026 }
00027
00028 NazaDecoder::FixType NazaDecoder::getFixType() {
00029     return fix;
00030 }
00031
00032 uint8_t NazaDecoder::getNumSat() {
00033     return sat;
00034 }
00035
00036 double NazaDecoder::getHeading() {
00037     return heading;
00038 }
00039
00040 double NazaDecoder::getCog() {
00041     return cog;
00042 }
00043
00044 double NazaDecoder::getGpsVsi() {
00045     return gpsVsi;
00046 }
00047
00048 double NazaDecoder::getHdop() {
00049     return hdop;
00050 }
00051
00052 double NazaDecoder::getVdop() {
00053     return vdop;
00054 }
00055
00056 uint8_t NazaDecoder::getYear() {
00057     return year;
00058 }
00059
00060 uint8_t NazaDecoder::getMonth() {
00061     return month;
00062 }
00063
00064 uint8_t NazaDecoder::getDay() {
00065     return day;
00066 }
00067
00068 uint8_t NazaDecoder::getHour() {
00069     return hour;
00070 }
00071
00072 uint8_t NazaDecoder::getMinute() {
00073     return minute;
00074 }
00075
00076 uint8_t NazaDecoder::getSecond() {
00077     return second;
00078 }
00079
00080 NazaDecoder::VersionType NazaDecoder::getFirmwareVersion
      () {
00081     return firmwareVersion;
00082 }
00083
00084 NazaDecoder::VersionType NazaDecoder::getHardwareVersion
      () {
00085     return hardwareVersion;
00086 }
00087
00088 uint8_t NazaDecoder::isLocked() {
00089     return locked;
00090 }
00091
```

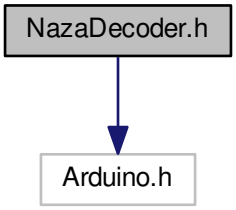
```
00092 uint8_t NazaDecoder::decode(int16_t input) {
00093
00094     // header (part 1 - 0x55)
00095     if ((seq == 0) && (input == 0x55)) {
00096         seq++;
00097     }
00098
00099     // header (part 2 - 0xaa)
00100     else if ((seq == 1) && (input == 0xaa)) {
00101         cs1 = 0;
00102         cs2 = 0;
00103         seq++;
00104     } else if (seq == 2) {
00105         msgId = input;
00106         updateChecksum(input);
00107         seq++;
00108     }
00109
00110     // message id
00111     // message payload length (should match message id)
00112     // store payload in buffer
00113     else if ((seq == 3) && (((msgId == NAZA_MESSAGE_GPS_TYPE) && (input ==
      NAZA_MESSAGE_GPS_SIZE)) || ((msgId ==
      NAZA_MESSAGE_COMPASS_TYPE) && (input ==
      NAZA_MESSAGE_COMPASS_SIZE)) || ((msgId ==
      NAZA_MESSAGE_MODULE_VERSION_TYPE) && (input ==
      NAZA_MESSAGE_MODULE_VERSION_SIZE)))) {
00114         msgLen = input;
00115         cnt = 0;
00116         updateChecksum(input);
00117         seq++;
00118     } else if (seq == 4) {
00119         payload[cnt++] = input;
00120         updateChecksum(input);
00121         if (cnt >= msgLen) {
00122             seq++;
00123         }
00124     }
00125
00126     // verify checksum #1
00127     else if ((seq == 5) && (input == cs1)) {
00128         seq++;
00129     }
00130
00131     // verify checksum #2
00132     else if ((seq == 6) && (input == cs2)) {
00133         seq++;
00134     } else {
00135         seq = 0;
00136     }
00137
00138     // all data in buffer
00139     if (seq == 7) {
00140         seq = 0;
00141
00142         // Decode GPS data
00143         if (msgId == NAZA_MESSAGE_GPS_TYPE) {
00144             uint8_t mask = payload[NAZA_MESSAGE_POS_XM];
00145             uint32_t time = pack4(NAZA_MESSAGE_POS_DT, mask);
00146             second = time & 0x3f;
00147             time >>= 6;
00148             minute = time & 0x3f;
00149             time >>= 6;
00150             hour = time & 0x0f;
00151             time >>= 4;
00152             day = time & 0x1f;
00153             time >>= 5;
00154             if (hour > 7) {
00155                 day++;
00156             }
00157             month = time & 0x0f;
00158             time >>= 4;
00159             year = time & 0x7f;
00160             lon = (double) pack4(NAZA_MESSAGE_POS_LO, mask) / 10000000;
00161             lat = (double) pack4(NAZA_MESSAGE_POS_LA, mask) / 10000000;
00162             gpsAlt = (double) pack4(NAZA_MESSAGE_POS_AL, mask) / 1000;
00163             double nVel = (double) pack4(NAZA_MESSAGE_POS_NV, mask) / 100;
00164             double eVel = (double) pack4(NAZA_MESSAGE_POS_EV, mask) / 100;
00165             spd = sqrt(nVel * nVel + eVel * eVel);
00166             cog = atan2(eVel, nVel) * 180.0 / M_PI;
00167             if (cog < 0) {
00168                 cog += 360.0;
00169             }
00170             gpsVsi = -(double) pack4(NAZA_MESSAGE_POS_DV, mask) / 100;
00171             vdop = (double) pack2(NAZA_MESSAGE_POS_VD, mask) / 100;
00172             double ndop = (double) pack2(NAZA_MESSAGE_POS_ND, mask) / 100;
00173             double edop = (double) pack2(NAZA_MESSAGE_POS_ED, mask) / 100;
```

```
00174             hdop = sqrt(ndop * ndop + edop * edop);
00175             sat = payload[NAZA_MESSAGE_POS_NS];
00176             uint8_t fixType = payload[NAZA_MESSAGE_POS_FT] ^ mask;
00177             uint8_t fixFlags = payload[NAZA_MESSAGE_POS_SF] ^ mask;
00178             switch (fixType) {
00179             case 2:
00180                 fix = FIX_2D;
00181                 break;
00182             case 3:
00183                 fix = FIX_3D;
00184                 break;
00185             default:
00186                 fix = NO_FIX;
00187                 break;
00188             }
00189             if ((fix != NO_FIX) && (fixFlags & 0x02)) {
00190                 fix = FIX_DGPS;
00191             }
00192             uint16_t lock = pack2(NAZA_MESSAGE_POS_SN, 0x00);
00193             locked = (lock == lastLock + 1);
00194             lastLock = lock;
00195         }
00196 
00197         // Decode compass data (not tilt compensated)
00198         // To calculate the heading (not tilt compensated) you need to do atan2 on the resulting y any a
      values, convert radians to degrees and add 360 if the result is negative.
00199         else if (msgId == NAZA_MESSAGE_COMPASS_TYPE) {
00200             uint8_t mask = payload[4];
00201             mask = (((mask ^ (mask >> 4)) & 0x0F) | ((mask << 3) & 0xF0)) ^ (((mask & 0x01) << 3) | ((mask
      & 0x01) << 7));
00202             int16_t x = pack2(NAZA_MESSAGE_POS_CX, mask);
00203             int16_t y = pack2(NAZA_MESSAGE_POS_CY, mask);
00204             if (x > magXMax) {
00205                 magXMax = x;
00206             }
00207             if (x < magXMin) {
00208                 magXMin = x;
00209             }
00210             if (y > magYMax) {
00211                 magYMax = y;
00212             }
00213             if (y < magYMin) {
00214                 magYMin = y;
00215             }
00216             heading = -atan2(y - ((magYMax + magYMin) / 2), x - ((
      magXMax + magXMin) / 2)) * 180.0 / M_PI;
00217             if (heading < 0) {
00218                 heading += 360.0;
00219             }
00220         } else if (msgId == NAZA_MESSAGE_MODULE_VERSION_TYPE) {
00221             firmwareVersion.version = pack4(
      NAZA_MESSAGE_POS_FW, 0x00);
00222             hardwareVersion.version = pack4(
      NAZA_MESSAGE_POS_HW, 0x00);
00223         }
00224         return msgId;
00225     } else {
00226         return NAZA_MESSAGE_NONE_TYPE;
00227     }
00228 }
00229 
00230 int32_t NazaDecoder::pack4(uint8_t i, uint8_t mask) {
00231     union {
00232         uint32_t d;
00233         uint8_t b[4];
00234     } v;
00235     for (int j = 0; j < 4; j++)
00236         v.b[j] = payload[i + j] ^ mask;
00237     return v.d;
00238 }
00239 
00240 int16_t NazaDecoder::pack2(uint8_t i, uint8_t mask) {
00241     union {
00242         uint16_t d;
00243         uint8_t b[2];
00244     } v;
00245     for (int j = 0; j < 2; j++) {
00246         v.b[j] = payload[i + j] ^ mask;
00247     }
00248     return v.d;
00249 }
00250 
```

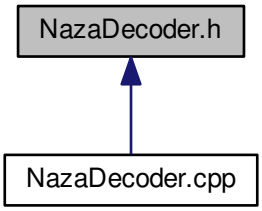
4.3 NazaDecoder.h File Reference

```
#include <Arduino.h>
```

Include dependency graph for NazaDecoder.h:



This graph shows which files directly or indirectly include this file:



Classes

- class NazaDecoder
- struct NazaDecoder::VersionSchemeType
- union NazaDecoder::VersionType

Macros

- #define MESSAGE_HEADER_SIZE 0x04

4.3.1 Macro Definition Documentation

4.3.1.1 #define MESSAGE_HEADER_SIZE 0x04

Arduino Naza Decoder.

Inspired by the Pawelsky's work.

Definition at line 12 of file NazaDecoder.h.

4.4 NazaDecoder.h

```
00001
00007 #ifndef __ARDUINO_NAZA_DECODER_H__
00008 #define __ARDUINO_NAZA_DECODER_H__
00009
00010 #include <Arduino.h>
00011
00012 #define MESSAGE_HEADER_SIZE 0x04
00013
00014 class NazaDecoder {
00015
00016 public:
00017
00018     typedef enum {
00019
00020         // date and time
00021         NAZA_MESSAGE_POS_DT = 0x04 - MESSAGE_HEADER_SIZE,
00022
00023         // longitude (x10^7, degree decimal)
00024         NAZA_MESSAGE_POS_LO = 0x08 - MESSAGE_HEADER_SIZE,
00025
00026         // latitude (x10^7, degree decimal)
00027         NAZA_MESSAGE_POS_LA = 0x0c - MESSAGE_HEADER_SIZE,
00028
00029         // altitude (in milimeters)
00030         NAZA_MESSAGE_POS_AL = 0x10 - MESSAGE_HEADER_SIZE,
00031
00032         // horizontal accuracy estimate (see uBlox NAV-POSLLH message for details)
00033         NAZA_MESSAGE_POS_HA = 0x14 - MESSAGE_HEADER_SIZE,
00034
00035         // vertical accuracy estimate (see uBlox NAV-POSLLH message for details)
00036         NAZA_MESSAGE_POS_VA = 0x18 - MESSAGE_HEADER_SIZE,
00037
00038         // NED north velocity (see uBlox NAV-VELNED message for details)
00039         NAZA_MESSAGE_POS_NV = 0x20 - MESSAGE_HEADER_SIZE,
00040
00041         // NED east velocity (see uBlox NAV-VELNED message for details)
00042         NAZA_MESSAGE_POS_EV = 0x24 - MESSAGE_HEADER_SIZE,
00043
00044         // NED down velocity (see uBlox NAV-VELNED message for details)
00045         NAZA_MESSAGE_POS_DV = 0x28 - MESSAGE_HEADER_SIZE,
00046
00047         // position DOP (see uBlox NAV-DOP message for details)
00048         NAZA_MESSAGE_POS_PD = 0x2c - MESSAGE_HEADER_SIZE,
00049
00050         // vertical DOP (see uBlox NAV-DOP message for details)
00051         NAZA_MESSAGE_POS_VD = 0x2e - MESSAGE_HEADER_SIZE,
00052
00053         // northing DOP (see uBlox NAV-DOP message for details)
00054         NAZA_MESSAGE_POS_ND = 0x30 - MESSAGE_HEADER_SIZE,
00055
00056         //easting DOP (see uBlox NAV-DOP message for details)
00057         NAZA_MESSAGE_POS_ED = 0x32 - MESSAGE_HEADER_SIZE,
00058
00059         // number of satellites (not XORed)
00060         NAZA_MESSAGE_POS_NS = 0x34 - MESSAGE_HEADER_SIZE,
00061
00062         // fix type (0 - no lock, 2 - 2D lock, 3 - 3D lock, not sure if other values can be expected - see
      uBlox NAV-SOL message for details)
00063         NAZA_MESSAGE_POS_FT = 0x36 - MESSAGE_HEADER_SIZE,
00064
00065         // fix status flags (see uBlox NAV-SOL message for details)
00066         NAZA_MESSAGE_POS_SF = 0x38 - MESSAGE_HEADER_SIZE,
00067
00068         // XOR mask
00069         NAZA_MESSAGE_POS_XM = 0x3b - MESSAGE_HEADER_SIZE,
00070
00071         // sequence number (not XORed), once there is a lock - increases with every message. When the lock
      is lost later LSB and MSB are swapped with every message.
00072         NAZA_MESSAGE_POS_SN = 0x3c - MESSAGE_HEADER_SIZE,
00073
00074         // checksum, calculated the same way as for uBlox binary messages
00075         NAZA_MESSAGE_POS_CS = 0x3e - MESSAGE_HEADER_SIZE
00076     } GPSPayloadPosition;
00077
00078     typedef enum {
00079
00080         // compass X axis data (signed)
00081         NAZA_MESSAGE_POS_CX = 0x04 - MESSAGE_HEADER_SIZE,
00082
00083         // compass Y axis data (signed)
00084         NAZA_MESSAGE_POS_CY = 0x06 - MESSAGE_HEADER_SIZE,
00085
00086         // compass Z axis data (signed)
00087         NAZA_MESSAGE_POS_CZ = 0x08 - MESSAGE_HEADER_SIZE
00088     } CompassPayloadPosition;
```

```
00089
00090     typedef enum {
00091
00092         // firmware version
00093         NAZA_MESSAGE_POS_FW = 0x08 - MESSAGE_HEADER_SIZE,
00094
00095         // hardware id
00096         NAZA_MESSAGE_POS_HW = 0x0c - MESSAGE_HEADER_SIZE
00097     } ModuleVersionPayloadPosition;
00098
00099     typedef enum {
00100         NAZA_MESSAGE_NONE_TYPE = 0x00,
00101         NAZA_MESSAGE_GPS_TYPE = 0x10,
00102         NAZA_MESSAGE_COMPASS_TYPE = 0x20,
00103         NAZA_MESSAGE_MODULE_VERSION_TYPE = 0x30
00104     } MessageType;
00105
00106     typedef enum {
00107         NAZA_MESSAGE_GPS_SIZE = 0x3a,
00108         NAZA_MESSAGE_COMPASS_SIZE = 0x06,
00109         NAZA_MESSAGE_MODULE_VERSION_SIZE = 0x0c
00110     } MessageSize;
00111
00112     typedef enum {
00113         NO_FIX = 0,
00114         FIX_2D = 2,
00115         FIX_3D = 3,
00116         FIX_DGPS = 4
00117     } FixType;
00118
00119     typedef struct {
00120         uint8_t revision;
00121         uint8_t build;
00122         uint8_t minor;
00123         uint8_t major;
00124     } VersionSchemeType;
00125
00126     typedef union {
00127         uint32_t version;
00128         VersionSchemeType scheme;
00129     } VersionType;
00130
00131     NazaDecoder();
00132
00133     uint8_t decode(int16_t input);
00134     double getLat();
00135     double getLon();
00136     double getGpsAlt();
00137     double getSpeed();
00138     FixType getFixType();
00139     uint8_t getNumSat();
00140     double getHeading();
00141     double getCog();
00142     double getGpsVsi();
00143     double getHdop();
00144     double getVdop();
00145     uint8_t getYear();
00146     uint8_t getMonth();
00147     uint8_t getDay();
00148
00149     // Note that for time between 16:00 and 23:59 the hour returned from GPS module is actually 00:00 -
      7:59.
00150     uint8_t getHour();
00151     uint8_t getMinute();
00152     uint8_t getSecond();
00153
00154     // Note that you need to read version numbers backwards (02 01 00 06 means v6.0.1.2)
00155     VersionType getFirmwareVersion();
00156     VersionType getHardwareVersion();
00157
00158     uint8_t isLocked();
00159 private:
00160     int16_t payload[58];
00161     int16_t seq;
00162     int16_t cnt;
00163     int16_t msgId;
00164     int16_t msgLen;
00165
00166     // checksum #1
00167     uint8_t cs1;
00168
00169     // checksum #2
00170     uint8_t cs2;
00171     int16_t magXMin;
00172     int16_t magXMax;
00173     int16_t magYMin;
00174     int16_t magYMax;
```

```
00175
00176     // longitude in degree decimal
00177     double lon;
00178
00179     // latitude in degree decimal
00180     double lat;
00181
00182     // altitude in m (from GPS)
00183     double gpsAlt;
00184
00185     // speed in m/s
00186     double spd;
00187
00188     // fix type
00189     FixType fix;
00190
00191     // number of satellites
00192     uint8_t sat;
00193
00194     // heading (not tilt compensated) in degrees
00195     double heading;
00196
00197     // course over ground
00198     double cog;
00199
00200     // vertical speed indicator (from GPS) in m/s (a.k.a. climb speed)
00201     double gpsVsi;
00202
00203     // horizontal dilution of precision
00204     double hdop;
00205
00206     // vertical dilution of precision
00207     double vdop;
00208     uint8_t year;
00209     uint8_t month;
00210     uint8_t day;
00211     uint8_t hour;
00212     uint8_t minute;
00213     uint8_t second;
00214
00215     VersionType firmwareVersion;
00216     VersionType hardwareVersion;
00217
00218     uint16_t lastLock;
00219     uint8_t locked;
00220
00221     int32_t pack4(uint8_t i, uint8_t mask);
00222
00223     int16_t pack2(uint8_t i, uint8_t mask);
00224
00225     void updateChecksum(int16_t input);
00226 };
00227
00228 #endif /* __ARDUINO_NAZA_DECODER_H__ */
```

Index