Arduino Naza Decoder Driver

Generated by Doxygen 1.8.9.1

Mon Dec 28 2015 21:00:47

Contents

1 Class Index

1.1 Class List

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

NazaDecoder **2**

2 File Index

2.1 File List

Here is a list of all files with brief descriptions:

NazaDecoder.cpp **7**

NazaDecoder.h **10**

3 Class Documentation

3.1 NazaDecoder Class Reference

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

Public Types

- enum fixType { NO_FIX = 0, FIX_2D = 2, FIX_3D = 3, FIX_DGPS = 4 }

Public Member Functions

- NazaDecoder ()
- uint8_t decode (int16_t input)
- int32_t getLat ()
- int32_t getLon ()
- int32_t getGpsAlt ()
- int32_t getSpeed ()
- fixType getFixType ()
- uint8_t getNumSat ()
- int32_t getHeading ()
- int32_t getCog ()
- int32_t getGpsVsi ()
- int32_t getHdop ()
- int32_t getVdop ()
- uint8_t getYear ()
- uint8_t getMonth ()
- uint8_t getDay ()
- uint8_t getHour ()
- uint8_t getMinute ()
- uint8_t getSecond ()

Private Member Functions

- void pack (uint8_t i, uint8_t mask, uint8_t ∗result, uint8_t len)
- 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
- int32_t lon
- int32_t lat
- int32_t gpsAlt
- int32_t spd
- fixType fix

- uint8_t sat
- int32_t heading
- int32_t cog
- int32_t gpsVsi
- int32_t hdop
- int32_t vdop
- uint8_t year
- uint8_t month
- uint8_t day
- uint8_t hour
- uint8_t minute
- uint8_t second

3.1.1 Detailed Description

Definition at line 16 of file NazaDecoder.h.

3.1.2 Member Enumeration Documentation

3.1.2.1 enum NazaDecoder::fixType

Enumerator

NO_FIX

FIX_2D

FIX_3D

FIX_DGPS

Definition at line 19 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 71 of file NazaDecoder.cpp.

3.1.4.2 int32_t NazaDecoder::getCog ( )

Definition at line 40 of file NazaDecoder.cpp.

3.1.4.3 uint8_t NazaDecoder::getDay ( )

Definition at line 58 of file NazaDecoder.cpp.

3.1.4.4 NazaDecoder::fixType NazaDecoder::getFixType ( )

Definition at line 31 of file NazaDecoder.cpp.

3.1.4.5 int32_t NazaDecoder::getGpsAlt ( )

Definition at line 25 of file NazaDecoder.cpp.

3.1.4.6 int32_t NazaDecoder::getGpsVsi ( )

Definition at line 43 of file NazaDecoder.cpp.

3.1.4.7 int32_t NazaDecoder::getHdop ( )

Definition at line 46 of file NazaDecoder.cpp.

3.1.4.8 int32_t NazaDecoder::getHeading ( )

Definition at line 37 of file NazaDecoder.cpp.

3.1.4.9 uint8_t NazaDecoder::getHour ( )

Definition at line 61 of file NazaDecoder.cpp.

3.1.4.10 int32_t NazaDecoder::getLat ( )

Definition at line 19 of file NazaDecoder.cpp.

3.1.4.11 int32_t NazaDecoder::getLon ( )

Definition at line 22 of file NazaDecoder.cpp.

3.1.4.12 uint8_t NazaDecoder::getMinute ( )

Definition at line 64 of file NazaDecoder.cpp.

3.1.4.13 uint8_t NazaDecoder::getMonth ( )

Definition at line 55 of file NazaDecoder.cpp.

3.1.4.14 uint8_t NazaDecoder::getNumSat ( )

Definition at line 34 of file NazaDecoder.cpp.

3.1.4.15 uint8_t NazaDecoder::getSecond ( )

Definition at line 67 of file NazaDecoder.cpp.

3.1.4.16 int32_t NazaDecoder::getSpeed ( )

Definition at line 28 of file NazaDecoder.cpp.

3.1.4.17 int32_t NazaDecoder::getVdop ( )

Definition at line 49 of file NazaDecoder.cpp.

3.1.4.18 uint8_t NazaDecoder::getYear ( )

Definition at line 52 of file NazaDecoder.cpp.

3.1.4.19 void NazaDecoder::pack (uint8_t *i*, uint8_t *mask*, uint8_t ∗ *result*, uint8_t *len*) `[private]`

Definition at line 8 of file NazaDecoder.cpp.

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

Definition at line 14 of file NazaDecoder.cpp.

3.1.5 Member Data Documentation

3.1.5.1 int16_t NazaDecoder::cnt [private]

Definition at line 52 of file NazaDecoder.h.

3.1.5.2 int32_t NazaDecoder::cog [private]

Definition at line 88 of file NazaDecoder.h.

3.1.5.3 uint8_t NazaDecoder::cs1 [private]

Definition at line 57 of file NazaDecoder.h.

3.1.5.4 uint8_t NazaDecoder::cs2 [private]

Definition at line 60 of file NazaDecoder.h.

3.1.5.5 uint8_t NazaDecoder::day [private]

Definition at line 100 of file NazaDecoder.h.

3.1.5.6 fixType NazaDecoder::fix [private]

Definition at line 79 of file NazaDecoder.h.

3.1.5.7 int32_t NazaDecoder::gpsAlt [private]

Definition at line 73 of file NazaDecoder.h.

3.1.5.8 int32_t NazaDecoder::gpsVsi [private]

Definition at line 91 of file NazaDecoder.h.

3.1.5.9 int32_t NazaDecoder::hdop [private]

Definition at line 94 of file NazaDecoder.h.

3.1.5.10 int32_t NazaDecoder::heading [private]

Definition at line 85 of file NazaDecoder.h.

3.1.5.11 uint8_t NazaDecoder::hour [private]

Definition at line 101 of file NazaDecoder.h.

3.1.5.12 int32_t NazaDecoder::lat [private]

Definition at line 70 of file NazaDecoder.h.

3.1.5.13 int32_t NazaDecoder::lon [private]

Definition at line 67 of file NazaDecoder.h.

3.1.5.14 int16_t NazaDecoder::magXMax [private]

Definition at line 62 of file NazaDecoder.h.

3.1.5.15 int16_t NazaDecoder::magXMin [private]

Definition at line 61 of file NazaDecoder.h.

3.1.5.16 int16_t NazaDecoder::magYMax [private]

Definition at line 64 of file NazaDecoder.h.

3.1.5.17 int16_t NazaDecoder::magYMin [private]

Definition at line 63 of file NazaDecoder.h.

3.1.5.18 uint8_t NazaDecoder::minute [private]

Definition at line 102 of file NazaDecoder.h.

3.1.5.19 uint8_t NazaDecoder::month [private]

Definition at line 99 of file NazaDecoder.h.

3.1.5.20 int16_t NazaDecoder::msgId [private]

Definition at line 53 of file NazaDecoder.h.

3.1.5.21 int16_t NazaDecoder::msgLen [private]

Definition at line 54 of file NazaDecoder.h.

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

Definition at line 50 of file NazaDecoder.h.

3.1.5.23 uint8_t NazaDecoder::sat [private]

Definition at line 82 of file NazaDecoder.h.

3.1.5.24 uint8_t NazaDecoder::second [private]

Definition at line 103 of file NazaDecoder.h.

3.1.5.25 int16_t NazaDecoder::seq [private]

Definition at line 51 of file NazaDecoder.h.

3.1.5.26 int32_t NazaDecoder::spd [private]

Definition at line 76 of file NazaDecoder.h.

3.1.5.27 int32_t NazaDecoder::vdop [private]

Definition at line 97 of file NazaDecoder.h.

3.1.5.28 uint8_t NazaDecoder::year [private]

Definition at line 98 of file NazaDecoder.h.

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

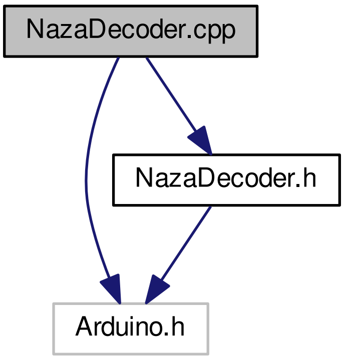
- NazaDecoder.h
- NazaDecoder.cpp

4 File Documentation

4.1 NazaDecoder.cpp File Reference

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

Include dependency graph for NazaDecoder.cpp:



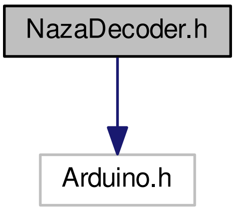
4.2 NazaDecoder.cpp

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

NazaDecoder::NazaDecoder()
        : 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) {
}

void NazaDecoder::pack(uint8_t i, uint8_t mask, uint8_t *result, uint8_t len) {
    for (int16_t j = 0; j < len; j++) {
        *(result++) = payload[i + j] ^ mask;
    }
}

void NazaDecoder::updateChecksum(int16_t input) {
    cs1 += input;
    cs2 += cs1;
}

int32_t NazaDecoder::getLat() {
    return lat;
}
int32_t NazaDecoder::getLon() {
    return lon;
}
int32_t NazaDecoder::getGpsAlt() {
    return gpsAlt;
}
int32_t NazaDecoder::getSpeed() {
    return spd;
}
NazaDecoder::fixType NazaDecoder::getFixType() {
    return fix;
}
uint8_t NazaDecoder::getNumSat() {
    return sat;
}
int32_t NazaDecoder::getHeading() {
    return heading;
}
```

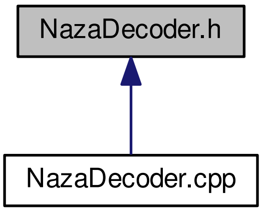
```
int32_t NazaDecoder::getCog() {
    return cog;
}
int32_t NazaDecoder::getGpsVsi() {
    return gpsVsi;
}
int32_t NazaDecoder::getHdop() {
    return hdop;
}
int32_t NazaDecoder::getVdop() {
    return vdop;
}
uint8_t NazaDecoder::getYear() {
    return year;
}
uint8_t NazaDecoder::getMonth() {
    return month;
}
uint8_t NazaDecoder::getDay() {
    return day;
}
uint8_t NazaDecoder::getHour() {
    return hour;
}
uint8_t NazaDecoder::getMinute() {
    return minute;
}
uint8_t NazaDecoder::getSecond() {
    return second;
}

uint8_t NazaDecoder::decode(int16_t input) {

    // header (part 1 - 0x55)
    if ((seq == 0) && (input == 0x55)) {
        seq++;
    }

    // header (part 2 - 0xAA)
    else if ((seq == 1) && (input == 0xAA)) {
        cs1 = 0;
        cs2 = 0;
        seq++;
    } else if (seq == 2) {
        msgId = input;
        updateChecksum(input);
        seq++;
    }

    // message id
    // message payload length (should match message id)
    // store payload in buffer
    else if ((seq == 3) && (((msgId == 0x10) && (input == 0x3A)) || ((msgId == 0x20) && (input == 0x06)))) {
        msgLen = input;
        cnt = 0;
        updateChecksum(input);
        seq++;
    } else if (seq == 4) {
        payload[cnt++] = input;
        updateChecksum(input);
        if (cnt >= msgLen) {
            seq++;
        }
    }

    // verify checksum #1
    else if ((seq == 5) && (input == cs1)) {
        seq++;
    }

    // verify checksum #2
    else if ((seq == 6) && (input == cs2)) {
        seq++;
    } else {
        seq = 0;
    }

    // all data in buffer
    if (seq == 7) {
        seq = 0;
        // Decode GPS data
        if (msgId == NAZA_MESSAGE_GPS) {
            uint8_t mask = payload[55];
            uint32_t time;
            pack(0, mask, (uint8_t *) &time, 4);
            second = time & 0x3f;
            time >>= 6;
            minute = time & 0x3f;
            time >>= 6;
            hour = time & 0x0f;
            time >>= 4;
            day = time & 0x1f;
            time >>= 5;
            if (hour > 7) {
                day++;
            }
            month = time & 0x0f;
            time >>= 4;
            year = time & 0x7f;

            uint32_t aux;
            pack(4, mask, (uint8_t *) &aux, 4);
            lon = (int32_t) aux / 10000000;
            pack(8, mask, (uint8_t *) &aux, 4);
            lat = (int32_t) aux / 10000000;
            pack(12, mask, (uint8_t *) &aux, 4);
            gpsAlt = (int32_t) aux / 1000;
            pack(28, mask, (uint8_t *) &aux, 4);
            int32_t nVel = (int32_t) aux / 100;
            pack(32, mask, (uint8_t *) &aux, 4);
            int32_t eVel = (int32_t) aux / 100;
            spd = sqrt(nVel * nVel + eVel * eVel);
            cog = atan2(eVel, nVel) * 180.0 / M_PI;
            if (cog < 0) {
                cog += 360.0;
            }
            pack(36, mask, (uint8_t *) &aux, 4);
            gpsVsi = -(int32_t) aux / 100;
            pack(42, mask, (uint8_t *) &aux, 4);
            vdop = (int32_t) aux / 100;
            pack(44, mask, (uint8_t *) &aux, 4);
            int32_t ndop = (int32_t) aux / 100;
            pack(46, mask, (uint8_t *) &aux, 4);
            int32_t edop = (int32_t) aux / 100;
            hdop = sqrt(ndop * ndop + edop * edop);
            sat = payload[48];
            uint8_t type = payload[50] ^ mask;
            uint8_t flag = payload[52] ^ mask;
            switch (type) {
            case 2:
                fix = FIX_2D;
                break;
            case 3:
                fix = FIX_3D;
                break;
            default:
                fix = NO_FIX;
                break;
            }
            if ((fix != NO_FIX) && (flag & 0x02)) {
                fix = FIX_DGPS;
            }
        }

        // Decode compass data (not tilt compensated)
        else if (msgId == NAZA_MESSAGE_COMPASS) {
            uint8_t mask = payload[4];
            mask = (((mask ^ (mask >> 4)) & 0x0f) | ((mask << 3) & 0xf0)) ^ (((mask & 0x01) << 3) | ((mask & 0x01) << 7));
            int16_t x;
            pack(0, mask, (uint8_t *) &x, 2);
            int16_t y;
            pack(2, mask, (uint8_t *) &y, 2);
            if (x > magXMax) {
                magXMax = x;
            }
            if (x < magXMin) {
                magXMin = x;
            }
            if (y > magYMax) {
                magYMax = y;
            }
            if (y < magYMin) {
                magYMin = y;
            }
            heading = -atan2(y - ((magYMax + magYMin) / 2), x - ((magXMax + magXMin) / 2)) * 180.0 / M_PI;
            if (heading < 0) {
                heading += 360.0;
            }
        }
        return msgId;
    } else {
        return NAZA_MESSAGE_NONE;
    }
}
```

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

Macros

- #define NAZA_MESSAGE_NONE 0x00
- #define NAZA_MESSAGE_GPS 0x10
- #define NAZA_MESSAGE_COMPASS 0x20

4.3.1 Macro Definition Documentation

4.3.1.1 #define NAZA_MESSAGE_COMPASS 0x20

Definition at line 14 of file NazaDecoder.h.

4.3.1.2 #define NAZA_MESSAGE_GPS 0x10

Definition at line 13 of file NazaDecoder.h.

4.3.1.3 #define NAZA_MESSAGE_NONE 0x00

Arduino Naza Decoder.

Inspired in 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 NAZA_MESSAGE_NONE    0x00
00013 #define NAZA_MESSAGE_GPS     0x10
00014 #define NAZA_MESSAGE_COMPASS 0x20
00015
00016 class NazaDecoder {
00017
00018 public:
00019     typedef enum {
00020         NO_FIX = 0,
00021         FIX_2D = 2,
00022         FIX_3D = 3,
00023         FIX_DGPS = 4
00024     } fixType;
00025
00026     NazaDecoder();
00027
00028     uint8_t decode(int16_t input);
00029     int32_t getLat();
00030     int32_t getLon();
00031     int32_t getGpsAlt();
00032     int32_t getSpeed();
00033     fixType getFixType();
00034     uint8_t getNumSat();
00035     int32_t getHeading();
00036     int32_t getCog();
00037     int32_t getGpsVsi();
00038     int32_t getHdop();
00039     int32_t getVdop();
00040     uint8_t getYear();
00041     uint8_t getMonth();
00042     uint8_t getDay();
00043
00044     // Note that for time between 16:00 and 23:59 the hour returned from GPS module is actually 00:00 -
       7:59.
00045     uint8_t getHour();
00046     uint8_t getMinute();
00047     uint8_t getSecond();
00048
00049 private:
00050     int16_t payload[58];
00051     int16_t seq;
00052     int16_t cnt;
00053     int16_t msgId;
00054     int16_t msgLen;
00055
00056     // checksum #1
00057     uint8_t cs1;
00058
00059     // checksum #2
00060     uint8_t cs2;
00061     int16_t magXMin;
00062     int16_t magXMax;
00063     int16_t magYMin;
00064     int16_t magYMax;
00065
00066     // longitude in degree decimal
00067     int32_t lon;
00068
00069     // latitude in degree decimal
00070     int32_t lat;
00071
```

```
    // altitude in m (from GPS)
    int32_t gpsAlt;

    // speed in m/s
    int32_t spd;

    // fix type
    fixType fix;

    // number of satellites
    uint8_t sat;

    // heading (not tilt compensated) in degrees
    int32_t heading;

    // course over ground
    int32_t cog;

    // vertical speed indicator (from GPS) in m/s (a.k.a. climb speed)
    int32_t gpsVsi;

    // horizontal dilution of precision
    int32_t hdop;

    // vertical dilution of precision
    int32_t vdop;
    uint8_t year;
    uint8_t month;
    uint8_t day;
    uint8_t hour;
    uint8_t minute;
    uint8_t second;

    void pack(uint8_t i, uint8_t mask, uint8_t *result, uint8_t len);
    void updateChecksum(int16_t input);
};

#endif /* __ARDUINO_NAZA_DECODER_H__ */
```

Index