06/09/25 Wesley Cooke

Reading NAV-PVT on the ZEDF9P without a Library

also see page 41

GOAL

- [ ] Explore the UBX Protocol
- [ ] Request the NAV-PVT packet
- [ ] See the response of the NAV PVT packet

5.14.11 UBX-NAV-PVT (0x01 0x07)

5.14.11.1 Navigation Position Velocity Time Solution

Message	**UBX-NAV-PVT**
Description	**Navigation Position Velocity Time Solution**
Firmware	Supported on: • u-blox 9 with protocol version 27.11
Type	Periodic/Polled
Comment	Note that during a leap second there may be more or less than 60 seconds in a minute. See the section Leap seconds in Integration manual for details. This message combines position, velocity and time solution, including accuracy figures

Message Structure	Header	Class	ID	Length (Bytes)	Payload	Checksum
	0xB5 0x62	0x01	0x07	92	see below	CK_A CK_B

To request the NAV_PVT message, we have to send the above message with a length of 0 and no payload. (see page 90).

Example Code

- [ ] craft the packet
  - Compute the checksum (see page 89)
- [ ] Wait for GPSAvailable(); to have bytes ready (See Page 88)
- [ ] Read the data from the gps

Reading NAV-PVT on the ZEDF9P without a Library

06/09/25
Wesley Cook

```
21 void setup()
22 {
23   Wire.begin();        // Start the I2C bus
24   Serial.begin(9600); // Start serial comms
25 }
26
27 void loop()
28 {
29   // Message to send to request NAV_PVT.
30   // 0x01 is the Class
31   // 0x07 is the ID
32   // 0x00 and 0x00 is the length
33   // Per the data sheet: we send an empty packet to request the data.
34   uint8_t nav_pvt[4] = { 0x01, 0x07, 0x00, 0x00};
35
36   // Checksums
37   byte CK_A = 0;
38   byte CK_B = 0;
39
40   // Compute Checksum
41   for (int ii=0; ii<4; ii++)
42   {
43     CK_A = CK_A + nav_pvt[ii];
44     CK_B = CK_B + CK_A;
45   }
46
47   // Send the packet
48   Wire.beginTransmission(gps_add); ☐ Craft the packet
49   Wire.write(0xB5); // Sync Char1
50   Wire.write(0x62); // Sync Char2
51   Wire.write(0x01); // Class
52   Wire.write(0x07); // ID
53   Wire.write(0x00); // Length in Little Endian Order!
54   Wire.write(0x00);
55   Wire.write(CK_A); // Check sum!
56   Wire.write(CK_B);
57   Wire.endTransmission();
58
59   // Wait for the GPS to tell us it has a response
60   while(GPSAvailable() < 0)
61   {}
62
63   uint16_t availableBytes = GPSAvailable();
64
65   if (availableBytes == 100)
66   {
67     Serial.println("Got 100 Bytes!");
68     // We expect 100 bytes because there are 2 sync bytes, a class, an id, 2 length bytes, and 92
69     // 2 + 1 + 1 + 2 + 92 + 2 = 92 + 8 = 100!  // payload bytes and 2 checksum bytes.
70   }
71   else
72   {
73     Serial.println("Weird number of bytes...");
74   }
75
76   // Read the bytes available from the GPS and print them in HEX ☐ Read the data
77   Wire.requestFrom(gps_add, availableBytes);
78   for (int ii=0; ii<availableBytes; ii++)
79   {
80     Serial.print(Wire.read(), HEX); Serial.print(" ");
81   }
82   // new line between readings.
83   Serial.println();
84   delay(1500);
85 }
```

Example output