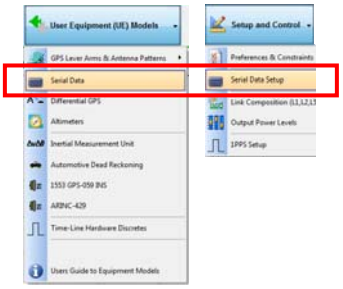
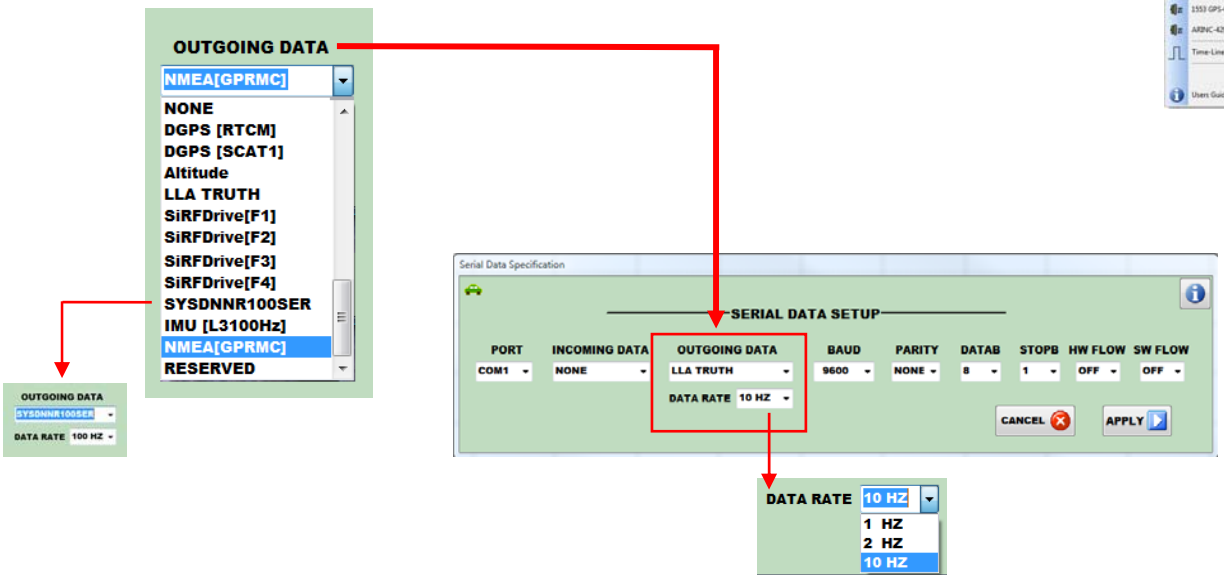
USING SERIAL DATA

Tapestry provides a selection of SERIAL DATA outputs that can used to enhance the overall simulation capability and provide a real-time continual data output.

Modify, enable, and construct Serial Data by using the **Serial Data Setup** Data Form accessed by one of the paths shown. Other, *model specific* Serial Data can be setup using the applicable **User Equipment Model**.





DATA FORMAT & CONTENT

- **DGPS [RTCM]:** Standard Differential GPS corrections in the 6-of-8 format as specified in the RTCM standard. This data can be connected directly to a serial data port on the Receiver under test. Use this link for more details.
- **DGPS [SCAT I]:** Special Category-I Differential GPS corrections defined for avionics. See this link for a definition of the format and further details.
- **ALTITUDE:** Ellipsoid Altitude as defined by WGS-84. To simulate MSL add the offset between Ellipsoid and Geod as an applied bias in the **Altimeter UE Model** selected from the **User Equipment** pulldown.

```
ALTIMETER: 3 OUTPUT TYPES

TYPE RMS:
        sprintf( SerialBuffer,"RMS %+06ldT-99",(long) altitude*FEET_PER_METER);
        // Checksum
        sum = 0;
        for ( i=0; i<14; i++ ) {  sum = sum + SerialBuffer[i]   }

        // write the checksum and finish the string
        sprintf( &SerialBuffer[14],"%02X%c\0",sum, 0x0d);


TYPE <ALT CR / LF>:

        sprintf( SerialBuffer,"%.1lf%c%c ", BaroAltitude, 0x0d, 0x0a);


TYPE <TIME, ALT CR / LF>:

        sprintf( SerialBuffer,"%.1lf,%.1lf%c%c ", SecondsIntoWeek, BaroAltitude, 0x0d, 0x0a);
```

- **LLA TRUTH:** Time tagged Latitude, Longitude, Altitude and other tangent frame output truth data is provided. Select from 1-10 Hz ASCII comma delimited output. The output format is a string of characters:

```
sprintf(    SerialBuffer,   "%4d,%8.1lf,%11.6lf,%11.6lf,%10.2lf,%8.3lf,%8.3lf,%8.3lf,%8.3lf,%8.3lf%c\0",
            NavData.Week,
            NavData.Time,                       // seconds into week
            NavData.Latitude*R2D,
            NavData.Longitude*R2D,
            NavData.Altitude,                   // Ellipsoid WGS84
            NavData.GroundSpeed,                // m/s
            NavData.ClimbRate,                  // m/s positive up
            NavData.Roll*R2D,
            NavData.Pitch*R2D,
            NavData.Heading*R2D,  0x0d);        // comma delimited followed by < CR >
```

- **SYSTRON-DONNER 100Hz IMU:** Data content as follows is provided at 100 Hz as a string of bytes defined as follows:

Header (0xFF81) + 6 words of Inertial Data + Checksum.

- Fixed Point, word 2 bytes / transmitted low byte first,
- Δv Scale Factor 2^{12} (meter/sec)
- $\Delta\theta$ Scale Factor 2^{17} (Degrees)
- Checksum $\equiv -(\Delta v_x + \Delta v_y + \Delta v_z + \Delta\theta_x + \Delta\theta_y + \Delta\theta_z)$
- Output Rate = 100 Hz / RS232

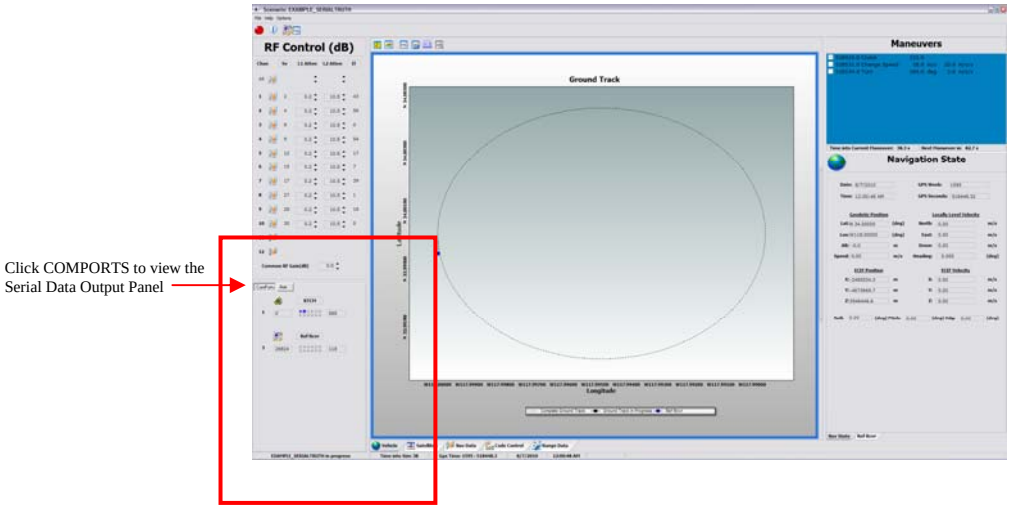
```
// Header
Header = 0xFF81;
fwrite(&Header, sizeof(short),1,hImu);          // 2 byte count

// Inertial Data
fwrite(&Δvz , sizeof(short) ,1, hImu);          // 4
fwrite(&Δθz , sizeof(short) ,1, hImu);          // 6
fwrite(&Δvy , sizeof(short) ,1, hImu);          // 8
fwrite(&Δθx , sizeof(short) ,1, hImu);          // 10
fwrite(&Δvx , sizeof(short) ,1, hImu);          // 12
fwrite(&Δθy , sizeof(short) ,1, hImu);          // 14

// Checksum
fwrite(&CheckSum,sizeof(short),1,hImu);     //  16
```

- **NMEA (MAP DATA):** NMEA data is provided suitable for direct input into a terrestrial mapping display program such as Map'N'Go, StreetPilot, etc. (*don't forget a NULL Modem*)

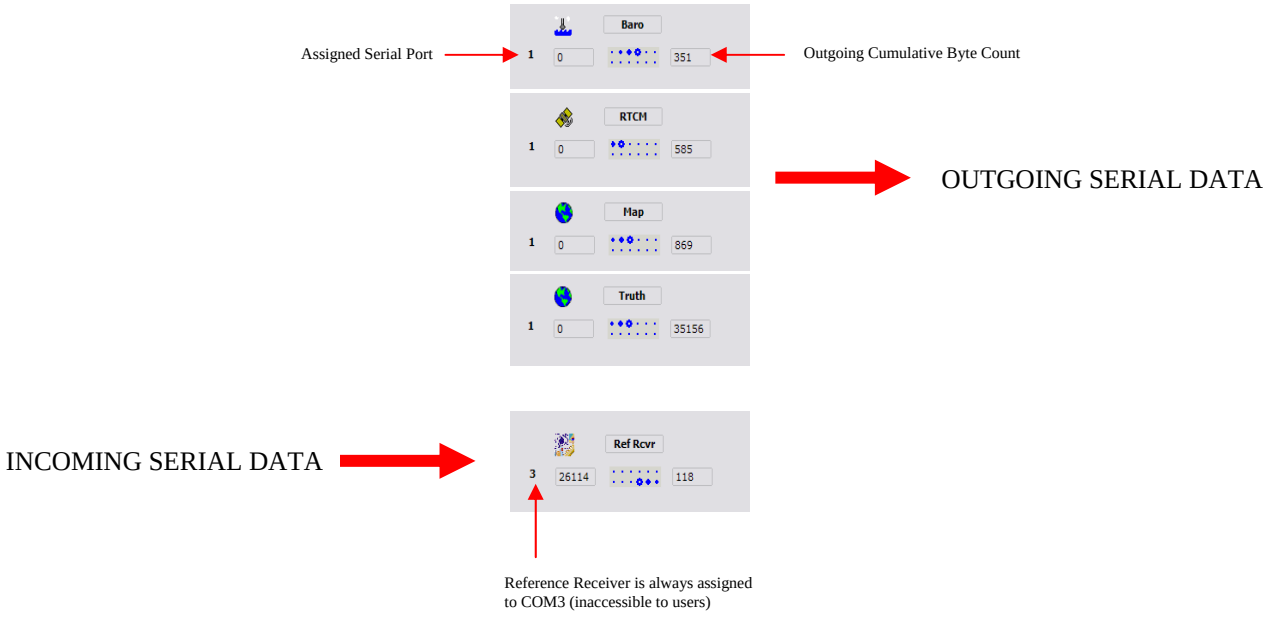
```
Year = Year - 2000;

sprintf(    SerialBuffer,"$GPRMC,%02d%02d%02d,A,%2d%.6lf,%c,%3d%.6lf,%c,0.0,%.3lf,%.3lf,%02d%02d%02d,000.0,E*\0",

            Hour, Minute, Second,
            LatDegree,  LatMinutes,  cNS,
            LonDegree, LonMinutes, cEW,
            Speed,  Heading*R2D, Year,Month,Day);

            // Checksum
            Sum  = 0;
            Num  = strlen(SerialBuffer);
            for (int I = 0; I < Num; i++ )  {  sum = sum + SerialBuffer[i]  }

sprintf( &SerialBuffer[Num],"%02X%c%c\0",sum, 0x0d, 0x0a);     // CR  LF
```

VERIFYING SERIAL OUTPUT

From the Run Scenario display, locate the Sensor Output Panel.



Active serial output will be reflected by an updating Byte count and one of the following ICONS depending upon the Data Type.



NOTE: You may capture the Serial Data using HyperTerminal. Generally for PC to PC communication, you'll need to use a **NULL** Modem