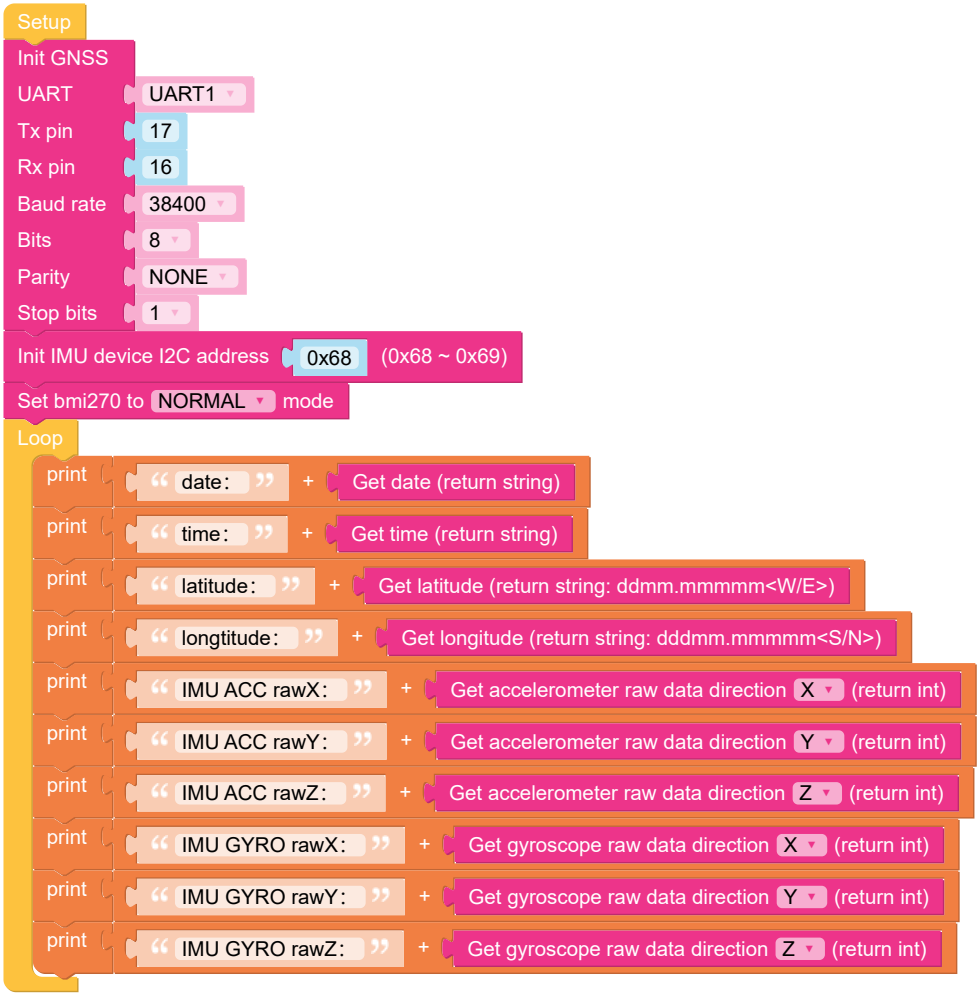
Module GNSS

Example

This program initializes the GNSS and IMU devices, retrieves the current date, time, latitude, and longitude information, and reads the raw data from the accelerometer and gyroscope on the X, Y, and Z axes, then continuously prints this information.



```
from m5stack import *
from m5ui import *
from uiflow import *
import module

setScreenColor(0x222222)

gnss = module.get(module.GNSS)

gnss.init_gnss(1, 17, 16, 38400, 8, None, 1)
gnss.init_imu(0x68)
gnss.set_mode('normal')
while True:
  print((str('date: ') + str((gnss.gnss_date))))
  print((str('time: ') + str((gnss.gnss_time))))
  print((str('latitude: ') + str((gnss.latitude))))
  print((str('longitude: ') + str((gnss.longitude))))
  print((str('IMU ACC rawX: ') + str((gnss.get_accel(1)[0]))))
  print((str('IMU ACC rawY: ') + str((gnss.get_accel(1)[1]))))
  print((str('IMU ACC rawZ: ') + str((gnss.get_accel(1)[2]))))
  print((str('IMU GYRO rawX: ') + str((gnss.get_gyro(1)[0]))))
  print((str('IMU GYRO rawY: ') + str((gnss.get_gyro(1)[1]))))
  print((str('IMU GYRO rawZ: ') + str((gnss.get_gyro(1)[2]))))
  wait_ms(2)
```

API

Get accelerometer data direction X (m/s2, return float)

```
gnss.get_accel(0)[0]
```

- Retrieves the accelerometer data in the specified direction (e.g., X-axis) in meters per second squared (m/s²), returning a floating-point number.

Get accelerometer raw data direction X (return int)

```
gnss.get_accel(1)[0]
```

- Retrieves the raw accelerometer data in the specified direction (e.g., X-axis), returning an integer value.

Get altitude (meter, return string)

```
gnss.altitude
```

- Retrieves the current altitude in meters, returning a string representation of the altitude.

Get course (return string)

```
gnss.course
```

- Retrieves the course information, returning a string representation of the course. The course refers to the direction relative to the Earth's North Pole.

Get date (return string)

```
gnss.gnss_date
```

- Retrieves the current date information, returning a string representation of the date.

Get gyroscope data direction X (deg/s, return float)

```
gnss.get_gyro(0)[0]
```

- Retrieves the gyroscope data in the specified direction (e.g., X-axis) in degrees per second (deg/s), returning a floating-point number.

Get gyroscope raw data direction X (return int)

```
gnss.get_gyro(1)[0]
```

- Retrieves the raw gyroscope data in the specified direction (e.g., X-axis), returning an integer value.

Get latitude (return string: ddmm.mmmmm<W/E>)

```
gnss.latitude
```

- Retrieves the latitude information, returning a string in degrees and minutes format (ddmm.mmmmm) with a W/E indicator for west or east longitude.

Get latitude decimal (return float:dd.dddd)

```
gnss.latitude_decimal
```

- Retrieves the latitude in decimal format, returning a floating-point number.

Get longitude (return string: dddmm.mmmmm<S/N>)

```
gnss.longitude
```

- Retrieves the longitude information, returning a string in degrees and minutes format (ddmm.mmmmm) with a S/N indicator for south or north longitude.

Get longitude decimal (return float:dd.dddd)

```
gnss.longitude_decimal
```

- Retrieves the longitude in decimal format, returning a floating-point number.

Get magnetometer magnetic field strength direction X (uT, return float)

```
gnss.get_magneto(0)[0]
```

- Retrieves the magnetometer data in the specified direction (e.g., X-axis) in microteslas (μT), returning a floating-point number.

Get magnetometer raw data direction X (return int)

```
gnss.get_magneto(1)[0]
```

- Retrieves the raw magnetometer data in the specified direction (e.g., X-axis), returning an integer value.

Get positioning quality (return string)

```
gnss.pos_quality
```

- Retrieves positioning quality information, returning a string. This is typically used to indicate the quality or accuracy of the GNSS signal.

Get pressure (hPa, return float)

```
gnss.get_pressure
```

- Retrieves the current pressure value from the pressure sensor in hectopascals (hPa), returning a floating-point number.

Get satellite num (return string)

```
gnss.satellite_num
```

- Retrieves the current number of connected satellites, returning a string.

Get speed knot (return string)

```
gnss.speed_knot
```

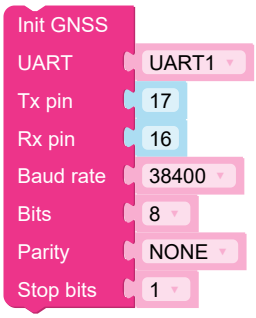
- Retrieves the speed, returning a string. The speed can be represented in knots (knot) or kilometers per hour (kph).


```
gnss.get_temperature
```

- Retrieves the current temperature in degrees Celsius (°C), returning a floating-point number.


```
gnss.gnss_time
```

- Retrieves the current time, returning a string representation of the time.




```
gnss.init_gnss(1, 17, 16, 38400, 8, None, 1)
```

- Initializes the GNSS module. Sets the UART communication parameters, including the UART port, Tx (transmit) pin, Rx (receive) pin, baud rate, data bits, parity, and stop bits.


```
gnss.init_imu(0x68)
```

- Initializes the IMU device with the I2C address set to 0x68, used for subsequent I2C communication.


```
gnss.set_acc_odr(0x09)
```

- Sets the accelerometer's output data rate. Here, 200Hz is selected, meaning the accelerometer outputs data 200 times per second.


```
gnss.set_acc_range(0x00)
```

- Sets the accelerometer's measurement range. Here, 2G is selected, meaning the accelerometer's maximum measurement range is ±2 times gravity (G).


```
gnss.set_mode('normal')
```

- Sets the BMI270 sensor to normal operating mode. The BMI270 is an IMU sensor typically used for attitude sensing and motion detection.


```
gnss.set_acc_odr(0x09)
```

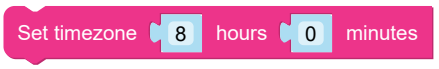
- Sets the gyroscope's output data rate. Here, 200Hz is selected, meaning the gyroscope outputs data 200 times per second.


```
gnss.set_gyr_range(0x01)
```

- Sets the gyroscope's measurement range. Here, 1000 degrees per second (dps) is selected, meaning the gyroscope's maximum measurement range is ±1000 degrees per second.


```
gnss.set_magneto_odr(0x00)
```

- Sets the magnetometer's output data rate. Here, 10Hz is selected, meaning the magnetometer outputs data 10 times per second.




```
gnss.set_time_zone(8)
```

- Sets the time zone. Here, it is set to 8 hours and 0 minutes, typically used to adjust the time to match the local time zone.