function Decode(fPort, bytes, variables) {

var obj = {};

var offset = 0;

if( (fPort < 10 || fPort > 27) && fPort != 197 && fPort != 198){

obj.msgType = "unknown";

obj.code = 1;

obj.error = "Unknown fPort,expect([10,25]),actual(" + fPort + ")";

return obj;

}

if(null == bytes){

return {};

}

var indexI, indexJ;

switch(fPort){

//It's a heartbeat message.

case 10:

{

obj.msgType = "heartbeat";

if(bytes.length < 16){

obj.code = 2;

obj.error = "Wrong message length,expect(16,18,20),actual(" + bytes.length + ")";

return obj;

}

obj.code = 0;

obj.version = {};

obj.version.swVer = ((bytes[0] >> 4) & 0x3).toString() + "." + (bytes[0] & 0xf).toString();

obj.version.hardwareType = (bytes[0] >> 6);// 1: "gateway" , 0: "badge";

obj.signal = {};

if (bytes[1] <= 127)

obj.signal.rssi = bytes[1] - 20;

else

obj.signal.rssi = bytes[1] - 275;

if(bytes[2] <= 127)

obj.signal.snr = bytes[2];

else

obj.signal.snr = bytes[2]- 255;

obj.status = {};

//GNSS status

obj.status.gnss = (bytes[3] >>> 5) & 0x3; //0:off,1:positioning,2:successful,3:failed

obj.status.battery = (bytes[3] >> 3) & 0x3; //0:not charging,1:charging,2:complete,3:unknown

//During the heartbeat period, whether the device is moved.

obj.status.vibstate = (bytes[3] >> 2) & 0x1;//1: moved,0:static;

/\*

Only used when the hardware type is badge.

0 indicates it works as a tracker

1 indicates it works as a BLE gateway

\*/

obj.workmode = (bytes[3] >> 1) & 0x1;//1:gateway,0:tracker;

//Calculate the available power of the battery.

var voltage = bytes[4]/ 100.0 + 2;

if(voltage < 3.3){

obj.status.soc = 0;

}

else if(voltage >= 4.15){

obj.status.soc = 100;

}

else if(voltage >= 4.1){

obj.status.soc = 99;

}

else if(voltage >= 4.04){

obj.status.soc = Math.round(96 + 3 \* (voltage - 4.04) / (4.1 - 4.04));

}

else if(voltage >= 3.97){

obj.status.soc = Math.round(90 + 6 \* (voltage - 3.97) / (4.04 - 3.97));

}

else if(voltage >= 3.92){

obj.status.soc = Math.round(69 + 21 \* (voltage - 3.92) / (3.97 - 3.92));

}

else if(voltage >= 3.69){

obj.status.soc =Math.round (25 + 44 \* (voltage - 3.69) / (3.92 - 3.69));

}

else if(voltage >= 3.4){

obj.status.soc = Math.round(3 + 22 \* (voltage - 3.4) / (3.69 - 3.4));

}

else if(voltage >= 3.3){

obj.status.soc = Math.round(3 \* (voltage - 3.3) / (3.4 - 3.3));

}

//Parse the parameters.

obj.parameters = {};

obj.parameters.txPower = bytes[5] >>> 6;

//Data rate

obj.parameters.dr = (bytes[5] >> 3) & 0x7;

//Scheme

// 0:US915,1:EU868,2:AU915,3:CN470,4:AS923,5:KR920,6:IN865,7:RU864;

obj.parameters.scheme = bytes[5] & 0x7;

obj.parameters.ble = {};

obj.parameters.ble.auReport = bytes[6] >>> 7;//1:enable,0:disable;

var bleperiod = [0,5,10,20,30,60,120,300,600,900,1200,1800,3600,7200,21600,43200];

obj.parameters.ble.period = bleperiod[(bytes[6] >> 3) & 0x0f];

var scan = [1,2,3,6,9,12,15,255];

obj.parameters.ble.scan = scan[bytes[6] & 0x7];

obj.parameters.ble.scale = (bytes[7] >> 6) & 0x3;

obj.parameters.ble.stepsOff = ((bytes[7] >> 3) & 0x7) \* 5;

obj.parameters.ble.bleOff = bytes[7] & 0x7;

obj.parameters.warning = {};

//buzzer 0:disable,1:enable;

obj.parameters.warning.buzzer = (bytes[8] >> 5) & 0x1;

obj.parameters.warning.vibrator = (bytes[8] >> 4) & 0x1;//1:enabled,0:disabled;

var distance = [2,4,6,8,10,15,255];

obj.parameters.warning.distance = distance[(bytes[8] >> 1) & 0x7];

obj.parameters.warning.proximity = bytes[8] & 0x1;//1:enabled,0:disabled;

//Gnss report period

var gnssperiod = [0,10,20,30,60,120,300,600,1800,3600,7200,10800,21600,43200];

obj.parameters.gnssPeriod = gnssperiod[(bytes[9] >> 4) & 0xf];

//Heartbeat report period

var heartbeatperiod = [60,300,600,1200,1800,3600,7200,21600,43200,86400,86400,86400,86400,86400,86400,86400];

obj.parameters.heartBeatPeriod = heartbeatperiod[bytes[9] & 0xf];

if(obj.version.swVer > "1.4"){

obj.parameters.sleepy = {};

obj.parameters.sleepy.start = ((bytes[10] & 0x3) << 3) | ((bytes[11] >> 5) & 0x7);

obj.parameters.sleepy.end = bytes[11] & 0x1f;

obj.parameters.sleepy.degree = (bytes[10] >> 2) & 0x7;

if(obj.parameters.sleepy.start != obj.parameters.sleepy.end){

obj.parameters.timestamp = ((bytes[12] & 0xff) << 24) | ((bytes[13] & 0xff) << 16) | ((bytes[14] & 0xff) << 8) | (bytes[15] & 0xff);

obj.bleAck = (bytes[16] >> 3) & 0x01;

obj.thres = bytes[16] & 0x07;

obj.steps = ((bytes[17] & 0xff) << 16) | ((bytes[18] & 0xff) << 8) | (bytes[19] & 0xff);

}

else{

obj.bleAck = (bytes[12] >> 3) & 0x01;

obj.thres = bytes[12] & 0x07;

obj.steps = ((bytes[13] & 0xff) << 16) | ((bytes[14] & 0xff) << 8) | (bytes[15] & 0xff);

}

}

else{

var dateInd = bytes[10] & 0x40;

obj.parameters.time = {};

if(dateInd){

obj.parameters.time.date = ((bytes[14] & 0xff) << 8) | (bytes[15] & 0xff);

}

obj.parameters.time.hour = ((bytes[11] & 0x1)<<4) | (bytes[12] >> 4);

obj.parameters.time.minute = ((bytes[12] & 0xf) << 2) | (bytes[13] >> 6);

obj.parameters.time.second = bytes[13] & 0x3f;

obj.parameters.sleepy = {};

obj.parameters.sleepy.start = ((bytes[10] & 0x7) << 2) | (bytes[11] >> 6);

obj.parameters.sleepy.end = (bytes[11] >> 1) & 0x1f;

obj.parameters.sleepy.degree = (bytes[10] >> 3) & 0x7;

//Threshold

if(dateInd){

obj.bleAck = (bytes[16] >> 3) & 0x01;

obj.thres = bytes[16] & 0x07;

obj.steps = ((bytes[17] & 0xff) << 16) | ((bytes[18] & 0xff) << 8) | (bytes[19] & 0xff);

}

else{

obj.bleAck = (bytes[14] >> 3) & 0x01;

obj.thres = bytes[14] & 0x07;

obj.steps = ((bytes[15] & 0xff) << 16) | ((bytes[16] & 0xff) << 8) | (bytes[17] & 0xff);

}

}

return obj;

}

case 11:

{

obj.msgType = "GNSS coordinate";

if(bytes.length != 9 && bytes.length != 25){

obj.code = 2;

obj.error = "Wrong message length,expect(9,25),actual(" + bytes.length + ")";

return obj;

}

obj.code = 0;

if(bytes.length == 9)

{

obj.longitude = {};

obj.longitude.orientation = bytes[0] & 0x80 ? "W" : "E";

var longi = (bytes[0] & 0x7f) << 24 | bytes[1] << 16 | bytes[2] << 8 | bytes[3];

obj.longitude.value = (parseInt(longi / 10000000) + (longi % 10000000) / 6000000.0).toFixed(7);

obj.latitude = {};

obj.latitude.orientation = bytes[4] & 0x80 ? "S" : "N";

var lati = (bytes[4] & 0x7f) << 24 | bytes[5] << 16 | bytes[6] << 8 | bytes[7];

obj.latitude.value = (parseInt(lati / 10000000) + (lati % 10000000) / 6000000.0).toFixed(7);

obj.time = bytes[8];

}

else if(bytes.length == 25)

{

obj.longitude = {};

obj.longitude.orientation = bytes[0] & 0x80 ? "W" : "E";

var longi = (bytes[0] & 0x7f) << 24 | bytes[1] << 16 | bytes[2] << 8 | bytes[3];

obj.longitude.value = (parseInt(longi / 10000000) + (longi % 10000000) / 6000000.0).toFixed(7);

obj.latitude = {};

obj.latitude.orientation = bytes[4] & 0x80 ? "S" : "N";

var lati = (bytes[4] & 0x7f) << 24 | bytes[5] << 16 | bytes[6] << 8 | bytes[7];

obj.latitude.value = (parseInt(lati / 10000000) + (lati % 10000000) / 6000000.0).toFixed(7);

obj.longitude1 = {};

obj.longitude1.orientation = bytes[8] & 0x80 ? "W" : "E";

longi = (bytes[8] & 0x7f) << 24 | bytes[9] << 16 | bytes[10] << 8 | bytes[11];

obj.longitude1.value = (parseInt(longi / 10000000) + (longi % 10000000) / 6000000.0).toFixed(7);

obj.latitude1 = {};

obj.latitude1.orientation = bytes[12] & 0x80 ? "S" : "N";

lati = (bytes[12] & 0x7f) << 24 | bytes[13] << 16 | bytes[14] << 8 | bytes[15];

obj.latitude1.value = (parseInt(lati / 10000000) + (lati % 10000000) / 6000000.0).toFixed(7);

obj.longitude2 = {};

obj.longitude2.orientation = bytes[16] & 0x80 ? "W" : "E";

longi = (bytes[16] & 0x7f) << 24 | bytes[17] << 16 | bytes[18] << 8 | bytes[19];

obj.longitude2.value = (parseInt(longi / 10000000) + (longi % 10000000) / 6000000.0).toFixed(7);

obj.latitude2 = {};

obj.latitude2.orientation = bytes[20] & 0x80 ? "S" : "N";

lati = (bytes[20] & 0x7f) << 24 | bytes[21] << 16 | bytes[22] << 8 | bytes[23];

obj.latitude2.value = (parseInt(lati / 10000000) + (lati % 10000000) / 6000000.0).toFixed(7);

obj.time = bytes[24];

}

return obj;

}

case 12:

{

obj.msgType = "BLE coordinate";

if(bytes.length < 7){

obj.code = 2;

obj.error = "wrong message length,less than minimum length(10)";

return obj;

}

obj.code = 0;

var majorLen,majorShift,beaconStart;

if(bytes[1] == 0)

{

obj.step = bytes[0] << 16 | bytes[1] << 8 | bytes[2];

//How many kinds of Major value of the beacons.

majorLen = bytes[3] & 0xf;

majorShift = 4 + majorLen;

beaconStart = 4;

}

else{

majorLen = bytes[0] & 0xf;

majorShift = 1 + majorLen;

beaconStart = 1;

obj.closecontact = (bytes[0] >> 4) & 0x1;

}

obj.beaconList = [];

for(indexI = 0; indexI < majorLen; indexI++){

var beaconNum = bytes[beaconStart + indexI];

var major = (bytes[majorShift] << 8 | bytes[majorShift + 1]).toString(16);

var minorShift = majorShift + 2;

for(indexJ = 0; indexJ < beaconNum; indexJ++){

//var minorShift = majorShift + 2 + indexJ \*3;

var beaconObj = {};

beaconObj.major = major;

beaconObj.minor = ((bytes[minorShift] << 8) | bytes[minorShift + 1]).toString(16);

beaconObj.type = (bytes[minorShift + 2] >> 5) & 0x3;//0:locator,1:asset,2:alarm,3:proximity

beaconObj.rssi = -(bytes[minorShift + 2] & 0x1f) - 59;

minorShift += 2;

if(bytes[minorShift] >> 7){

beaconObj.battery = bytes[minorShift + 1] & 0x7f;

minorShift += 1;

}

minorShift += 1;

obj.beaconList.push(beaconObj);

}

majorShift = minorShift;

}

return obj;

}

case 13:

{

obj.msgType = "alarm";

if(bytes.length != 2){

obj.code = 2;

obj.error = "Wrong message length,expect(2),actual(" + bytes.length + ")";

return obj;

}

obj.code = 0;

obj.msgId = bytes[0];

obj.ack = (bytes[1] >> 4) & 0x1;// 1:true,0:false;

//0:SOS,1:SOS dismissed,2:power off,3:BLE disable,4:LoRa disable,5:GPS disable,6:Enter hazardous area,7:Unknown

obj.alarm = bytes[1] & 0x7;

return obj;

}

case 14:

{

obj.msgType = "ack";

if(bytes.length != 2){

obj.code = 2;

obj.error = "Wrong message length,expect(2),actual(" + bytes.length + ")";

return obj;

}

obj.code = 0;

obj.msgId = bytes[0];

//0:succeed,1:parameter not supported,2:parameter out of range,3:unknown;

obj.result = bytes[1] & 0x3;

return obj;

}

case 15:

{

obj.msgType = "Positioning UUID list";

obj.code = 0;

if(bytes[0] == 0){

return {};

}

obj.uuidList = [];

for(indexI = 0; indexI < bytes[0]; indexI++){

var uuid = {};

uuid.index = bytes[1 + 17 \* indexI];

var str = "";

for(indexJ = 2 + 17 \* indexI; indexJ < 18 + 17 \* indexI; indexJ++){

var tmp = bytes[indexJ].toString(16);

if(tmp.length == 1){

tmp = "0" + tmp;

}

str += tmp;

}

uuid.uuid = str;

obj.uuidList.push(uuid);

}

return obj;

}

case 16:

{

obj.msgType = "Asset UUID list";

obj.code = 0;

if(bytes[0] == 0){

return {};

}

obj.uuidList = [];

for(indexI = 0; indexI < bytes[0]; indexI++){

var uuidAsset = {};

uuidAsset.index = bytes[1 + 17 \* indexI];

var strAsset = "";

for(indexJ = 2 + 17 \* indexI; indexJ < 18 + 17 \* indexI; indexJ++){

var tmpAsset = bytes[indexJ].toString(16);

if(tmpAsset.length == 1){

tmpAsset = "0" + tmpAsset;

}

strAsset += tmpAsset;

}

uuidAsset.uuid = strAsset;

obj.uuidList.push(uuidAsset);

}

return obj;

}

case 17:

{

obj.msgType = "Pass-through filter list";

obj.code = 0;

if(bytes[0] == 0){

return obj;

}

obj.filterList = [];

var pos = 1;

for(indexI = 0; indexI < bytes[0]; indexI++){

var filterPosPass = {};

filterPosPass.port = bytes[pos];

filterPosPass.start = bytes[pos+1];

filterPosPass.end = bytes[pos+2];

filterPosPass.filterStart = bytes[pos + 3];

filterPosPass.filterLen= bytes[pos+4];

var filterLen = bytes[pos+4];

var strPosPass = "";

for(indexJ = 0; indexJ < filterLen; indexJ++){

var tmpPosPass = bytes[pos+5+indexJ].toString(16);

if(tmpPosPass.length == 1){

tmpPosPass = "0" + tmpPosPass;

}

strPosPass += tmpPosPass;

}

pos = pos + 5 + filterLen;

filterPosPass.filter = strPosPass;

obj.filterList.push(filterPosPass);

}

return obj;

}

case 18:

{

obj.msgType = "History Beacon Config List";

obj.code = 0;

if(bytes[0] == 0){

return {};

}

obj.number = bytes[0];

obj.beaconList = [];

for(indexI = 0; indexI < bytes[0]; indexI++){

var beacon= {};

offset = 5 \* indexI;

beacon.index = bytes[1 + offset];

beacon.major = (bytes[2 + offset] << 8 | bytes[3 + offset]).toString(16);

beacon.minor = (bytes[4 + offset] << 8 | bytes[5 + offset]).toString(16);

obj.beaconList.push(beacon);

}

return obj;

}

case 19:

{

obj.msgType = "History Beacon Info List";

obj.code = 0;

if(bytes[0] == 0){

return {};

}

obj.number = bytes[0];

obj.beaconList = [];

for(indexI = 0; indexI < bytes[0]; indexI++){

var bea= {};

offset = 9 \* indexI;

bea.major = (bytes[1 + offset] << 8 | bytes[2 + offset]).toString(16);

bea.minor = (bytes[3 + offset] << 8 | bytes[4 + offset]).toString(16);

bea.rssi = -((bytes[5 + offset] & 0x1f) + 59);

bea.frmOff = (bytes[6 + offset] << 8 | bytes[7 + offset]);

bea.timeOff = (bytes[8 + offset] << 8 | bytes[9 + offset]);

obj.beaconList.push(bea);

}

return obj;

}

case 20:

{

obj.msgType = "History GNSS Info List";

obj.code = 0;

if(bytes[0] == 0){

return {};

}

obj.number = bytes[0];

obj.gnssList = [];

for(indexI = 0; indexI < bytes[0]; indexI++){

var gnss = {};

offset = 12 \* indexI;

gnss.longitude = {};

gnss.longitude.orientation = bytes[1+offset] & 0x80 ? "W" : "E";

var longiH = (bytes[1+offset] & 0x7f) << 24 | bytes[2+offset] << 16 | bytes[3+offset] << 8 | bytes[4+offset];

gnss.longitude.value = (parseInt(longiH / 10000000) + (longiH % 10000000) / 6000000.0).toFixed(7);

gnss.latitude = {};

gnss.latitude.orientation = bytes[5+offset] & 0x80 ? "S" : "N";

var latiH = (bytes[5+offset] & 0x7f) << 24 | bytes[6+offset] << 16 | bytes[7+offset] << 8 | bytes[8+offset];

gnss.latitude.value = (parseInt(latiH / 10000000) + (latiH % 10000000) / 6000000.0).toFixed(7);

gnss.frmOff = (bytes[9 + offset] << 8 | bytes[10 + offset]);

gnss.timeoff = (bytes[11 + offset] << 8 | bytes[12 + offset]);

obj.gnssList.push(gnss);

}

return obj;

}

case 21:

case 22:

case 23:

case 24:

case 25:

{

obj.msgType = "Pass-through data list";

obj.port = fPort;

obj.code = 0;

if(bytes[0] == 0){

return {};

}

obj.dataList = [];

var dataLen = (bytes.length - 1)/bytes[0];

for(indexI = 0; indexI < bytes[0]; indexI++){

var passData = {};

passData.index = indexI;

var strDataPass = "";

for(indexJ = 1 + dataLen \* indexI; indexJ < 1 + dataLen + dataLen \* indexI; indexJ++){

var tmpDataPass = bytes[indexJ].toString(16);

if(tmpDataPass.length == 1){

tmpDataPass = "0" + tmpDataPass;

}

strDataPass += tmpDataPass;

}

passData.payload = strDataPass;

obj.dataList.push(passData);

}

return obj;

}

case 26:

{

var ii = 0;

var len = bytes.length;

var out = {access\_points: []};

for (; ii < len ;) {

out.access\_points.push({macAddress: bytes.slice(ii, ii + 6), signalStrength: int8(bytes[ii+6])});

ii += 7;

}

return out;

}

case 27:

{

return {

"sat\_count": bytes[0],

"lr1110\_gnss": bytes.slice(2, bytes.length)

};

}

default:

return {};

}

return {};

}

// convert a byte value to signed int8

function int8(byte) {

var sign = byte & (1 << 7);

if (sign) {

return 0xFFFFFF00 | byte;

}

return byte;

}