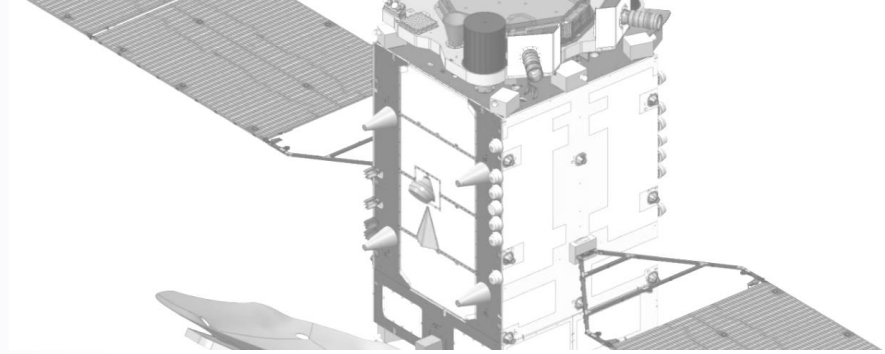


QZSS Emergency Warning Services (EWS) Trial

December 17, 2020

QZSS Strategy Office,
National Space Policy Secretariat
Cabinet Office, Government of Japan

Message Structure of DC Report service and EWS

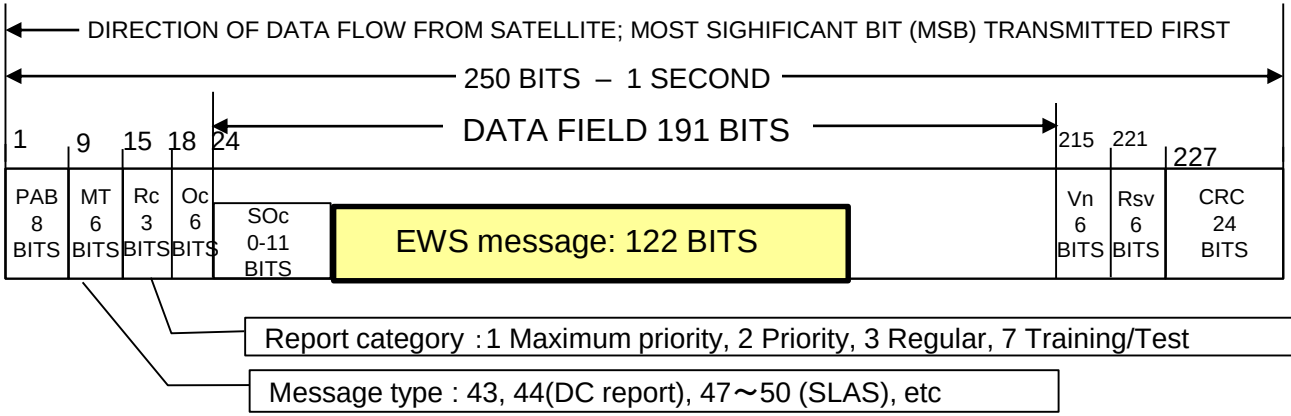
This page is the same as the last time.

- **Signal name : L1S (RF property and message structure are defined in IS-QZSS-L1S-004)**
- **Interval : DC report service : Once every 4 seconds**

  Sub-meter Level Augmentation Service : Twice every 4 seconds

Signal	Service Name	Center freq.	Modulation	Bit Rate
L1S	Sub-meter Level Augmentation Service (SLAS)	1575.42MHz	BPSK	250bps
	DC Report Service			

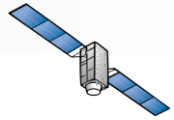
- **Message Structure**



PAB: Preamble
MT: Message type
Rc: Report Classification
Oc: Organization Code
Vn: Version Number
Rsv: Reserved
CRC: Cyclic Redundancy Check

	Message Type 43	Message Type 44
Outline	Disaster prevention information by Japan meteorological agency	Current: Arbitrary information Future: EWS
Contents	Information such as Earthquake, Tsunami, Volcano, etc.	Information delivered from external organization.
ICD	IS-QZSS-DCR-008	Not yet publiched Creating common EWS format with Europe

What you need to do for sending EWS message through QZSS



This page is the same as the last time.

1. Prepare EWS message data (122bits stream)
   - Prof. Shimazu (AIIT) will present “Common EWS format”
2. Provide EWS message data and test schedule via e-mail
   - The data should be provided before 10 working days for signal reception test and demonstration.
   - The test message transmission is allowed only during weekdays and day time (11:00-17:00JST / 09:00-15:00 Thai local time).
   - The number of messages should be within 30 kinds in a day.
   - One message is repeated once every four seconds during the requested test schedule. Test schedule is to be requested as “start time” and “end time”.
3. QSS, operating company of QZSS, checks and creates 250 bits stream for QZSS transmission message in advance of the test schedule.
   - NOTE: during one test message transmission, QZSS system adds different bit patterns on reserved and CRC bits to one EWS message you created. A message will become 48 different 250 bits message for the one 122 bits EWS message.
   - The test message patterns can be provided before the test for your validation.

Request flow for EWS message

This page is the same as the last time.

EWS message

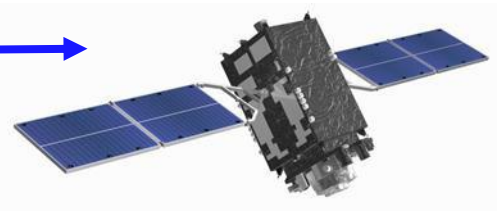
0100110110010011011001001101100100
1101100100110110010011011001001101
1001001101100100110110010011011O

122 bits

Send via e-mail before 10 days with test schedule

MT44

250 bits



You need to create.

Length in bits1	Content	Predefined value (integer)
8	Preamble	83 (A) or 154 (B) or 198 (C)
6	Message Type number	44
3	Report Classification	7 (test)
6	Organization Code	60 (foreign country)
7	Subdivision Org. Code	0
122	EWS message	Prepare this data (122bit)
62	Spare	0 (unused)
6	Version Number	0
6	Reserved	System operator will create
24	CRC	System operator will create

QZSS operator creates

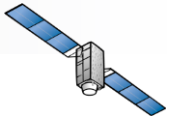
GPS Chip Antenna

Spresense

PC or Smartphone App etc…

Under RPD charrenge team scope

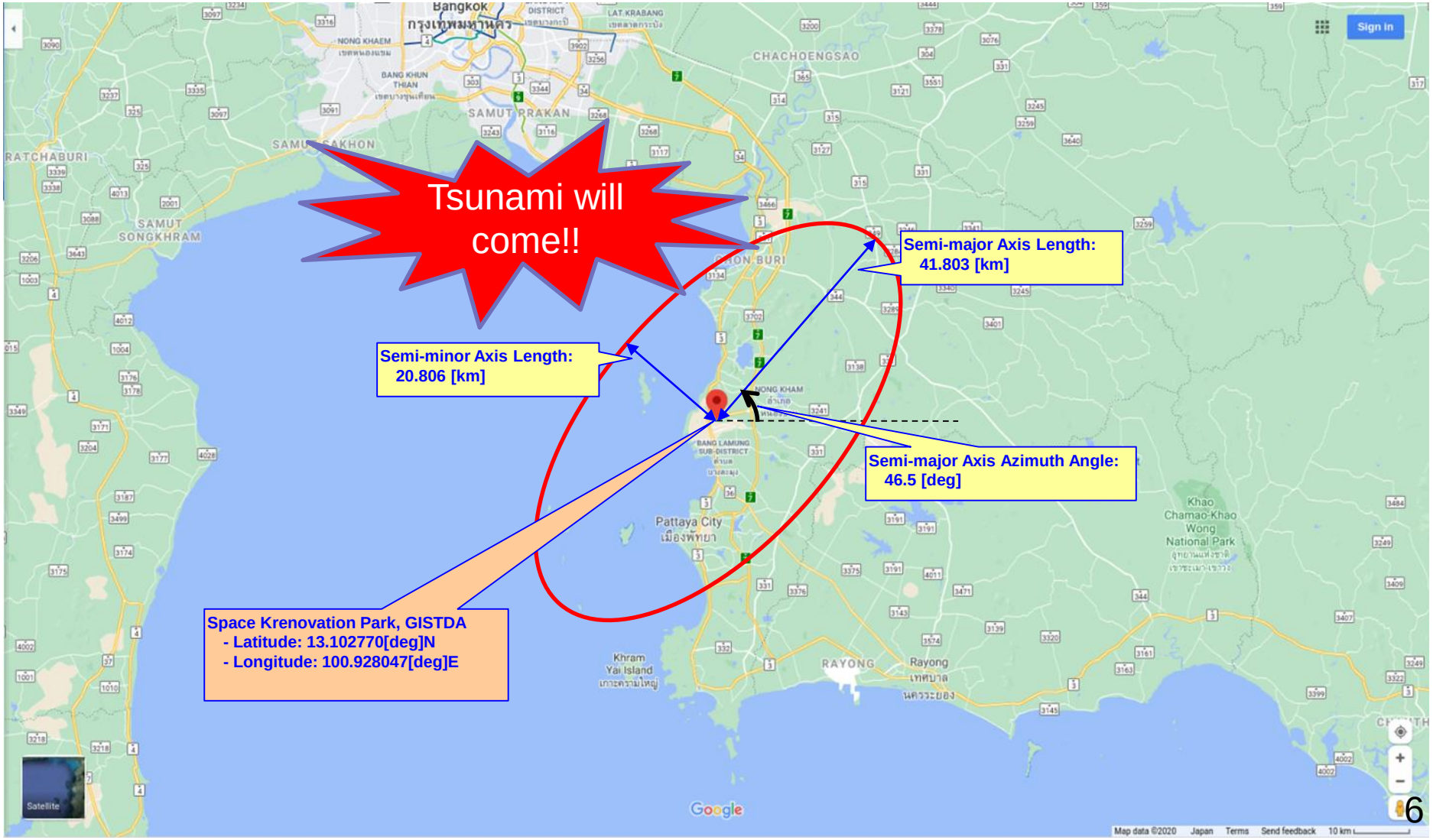
Under QZSS team scope



How you can make EWS message?

EWS message data (122bits)

Case1) Creating EWS message for Tsunami Alert in the following target area in Thailand.



EWS message data (122bits) for Case1

(1) Set up each parameter from EWS format(*).

(*) "EWS-Message(as of 08Sep2020)_for RPDchallenge.xlsx"

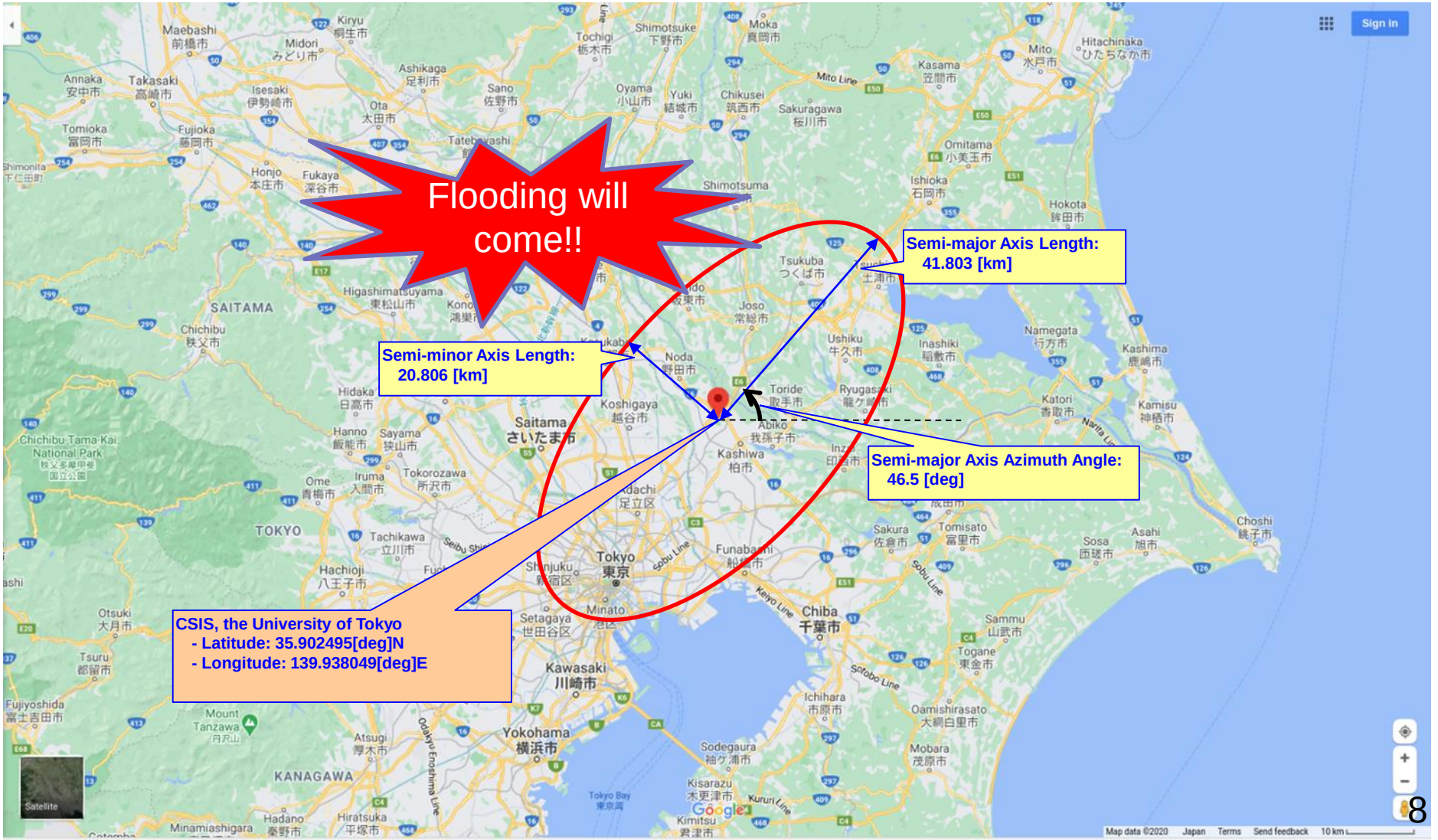
Message Field	Element Name	Binary Value	Description	Bit Length
Message Identifier	Message Type	10	Test	2
	Country ID	1011111100	Thailand(=764)	10
	Provider ID	0000	All 0	4
Event	Event Category	000	Geo	3
	Event Sub-Category	0001	Tsunami	4
	Severity	01	Severe	2
Event Chronology	Event Onset Day	00001	01 [day]	5
	Event Onset Hour	10111	23 [h]	5
	Event Onset Minute	101101	45 [min]	6
	Expected Duration	0000	No Duration	4
Guidance to React	Guidance Library	00	International guidance library	2
	Response type	0111	None	4
	Instructions	0000	Test	4
Target Area	Latitude	1001001010100010	13.102770[deg]N (1LSB=0.00275[deg])	16
	Longitude	11000111110001010	100.928047[deg]E (1LSB=0.00275)[deg])	17
	Semi-major Axis Length	0111	41803[m] (1LSB=316[m])	4
	Semi-minor Axis Length	0110	20806[m] (1LSB=316[m])	4
	Semi-major Axis Azimuth Angle	01000	46.45[deg] (1LSB=5.8[deg])	5
Parameters	Specific Setting	00000000000000000000	All 0	21
				122

(2) Based on the above setting, create EWS message with binary number as follows.

1010111111000000000000101000011011110110100000001110000100100101010001011000111110001010011101100100000000000000000000000000

EWS message data (122bits)

Case2) Creating EWS message for Flooding Alert in the following target area in Japan.



EWS message data (122bits) for Case2

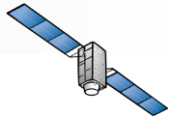
(1) Set up each parameter from EWS format(*).

(*) "EWS-Message(as of 08Sep2020)_for RPDchallenge.xlsx"

Message Field	Element Name	Binary Value	Description	Bit Length
Message Identifier	Message Type	10	Test	2
	Country ID	0110001000	Japan(=392)	10
	Provider ID	0000	All 0	4
Event	Event Category	011	Met	3
	Event Sub-Category	0001	Flood	4
	Severity	01	Severe	2
Event Chronology	Event Onset Day	00001	01 [day]	5
	Event Onset Hour	10111	23 [h]	5
	Event Onset Minute	101101	45 [min]	6
	Expected Duration	0000	No Duration	4
Guidance to React	Guidance Library	00	International guidance library	2
	Response type	0111	None	4
	Instructions	0000	Test	4
Target Area	Latitude	1011001100001111	35.902495[deg]N (1LSB=0.00275[deg])	16
	Longitude	11100011100000101	139.938049[deg]E (1LSB=0.00275)[deg])	17
	Semi-major Axis Length	0111	41803[m] (1LSB=316[m])	4
	Semi-minor Axis Length	0110	20806[m] (1LSB=316[m])	4
	Semi-major Axis Azimuth Angle	01000	46.45[deg] (1LSB=5.8[deg])	5
Parameters	Specific Setting	00000000000000000000	All 0	21
				122

(2) Based on the above setting, create EWS message with binary number as follows.

10011000100000000110001010000110111101101000000011100001011001100001111111000111000001010111011001000000000000000000000000



What data will be broadcasted from QZSS?

MT44 message data (250bits) for Case1 & 2

MT44 message data will be created by System operator as below.

The first 30bits are already fixed. And same bit stream is displayed in any EWS message.

30bits

Bit Length	Content		Predefined value (integer)
8	Preamble	**01010011**／ 10011010／ 11000110	83 (A) ／ 154 (B) ／ 198 (C)
6	Message Type number	**101100**	44
3	Report Classification	**111**	7 (test)
6	Organization Code	**111100**	60 (foreign country)
7	Subdivision Org. Code	**0000000**	0
122	EWS message	**101**01111110000000000010100001101111011010000000111000010010010101000101100011111000101001110110010000000000000000000000000	Prepare this data (122bit) **This will be created by each team.**
62	Spare		0 (unused)
6	Version Number		0
6	Reserved		System operator will create
24	CRC		System operator will create

01010011 101100 111 111100 0000000 10101111110000・・・

BIN	0101	0011	1011	0011	1111	1000	0000	0010	1011	1111	0000	・・・
HEX	5	3	B	3	F	8	0	2	B	F	0	・・・

MT44 message data (250bits) for Case1 & 2

The first 8bits of MT44 message are Preamble.

3types of following data are cycled through and displayed.

83 (A) → 154 (B) → 198 (C) → 83 (A) → 154 (B) → 198 (C) → ・・・

01010011 10011010 11000110 01010011 10011010 11000110

BIN	0101	0011	1011	0011	1111	1000	0000	0010	1011	1111	0000	・・・
HEX	5	3	B	3	F	8	0	2	B	F	0	・・・

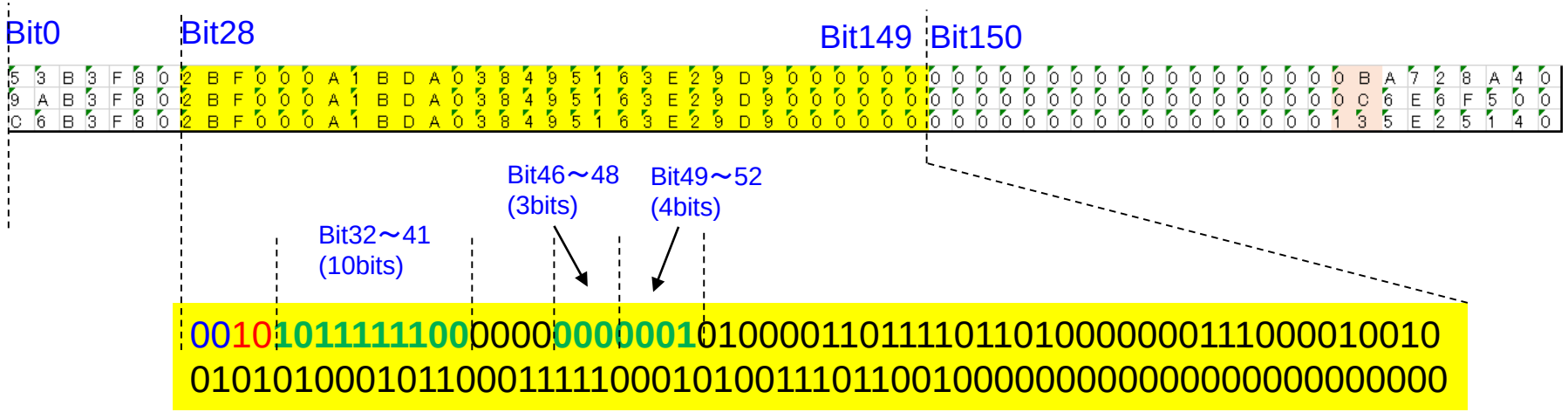
BIN	1001	1010	1011	0011	1111	1000	0000	0010	1011	1111	0000	・・・
HEX	9	A	B	3	F	8	0	2	B	F	0	・・・

BIN	1100	0110	1011	0011	1111	1000	0000	0010	1011	1111	0000	・・・
HEX	C	6	B	3	F	8	0	2	B	F	0	・・・

MT44 message data (250bits) to be displayed

You can extract some information as follows.

Case1)



Coumtry ID : Bit32 ～ Bit41 (10bits)

=>1011111100[BIN] = 764[DEC] = **Thailand**

Event Category : Bit46 ～ Bit48 (3bits)

=>000[BIN] = **Geo**

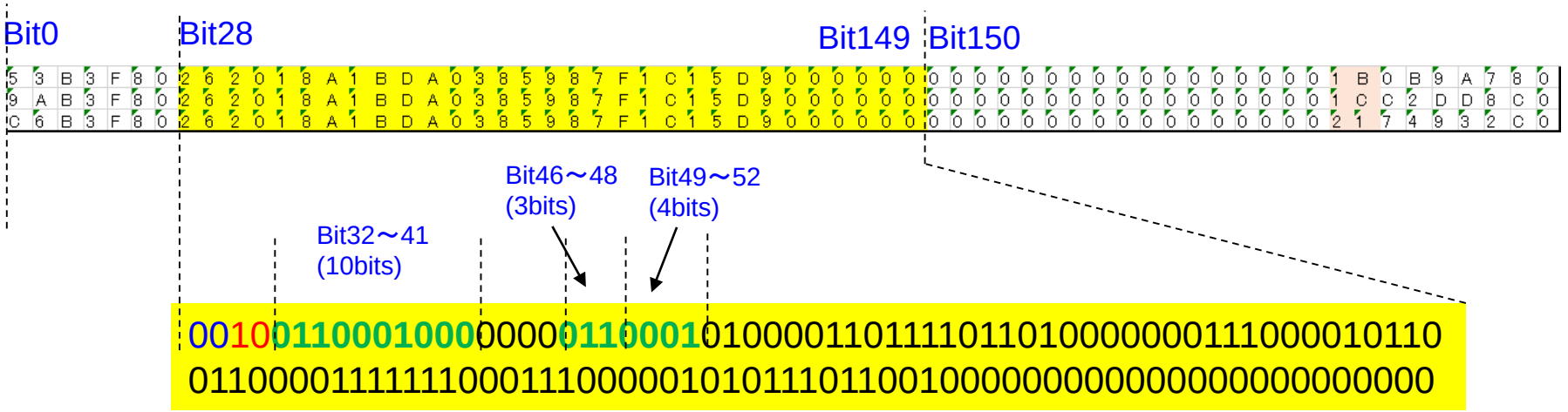
Event Sub Category : Bit49 ～ Bit52 (4bits)

=>0001[BIN] = **Tsunami**

MT44 message data (250bits) to be displayed

You can extract some information as follows.

Case2)



Coumtry ID : Bit32 ～ Bit41 (10bits)

=>0110001000[BIN] = 392[DEC] = **Japan**

Event Category : Bit46 ～ Bit48 (3bits)

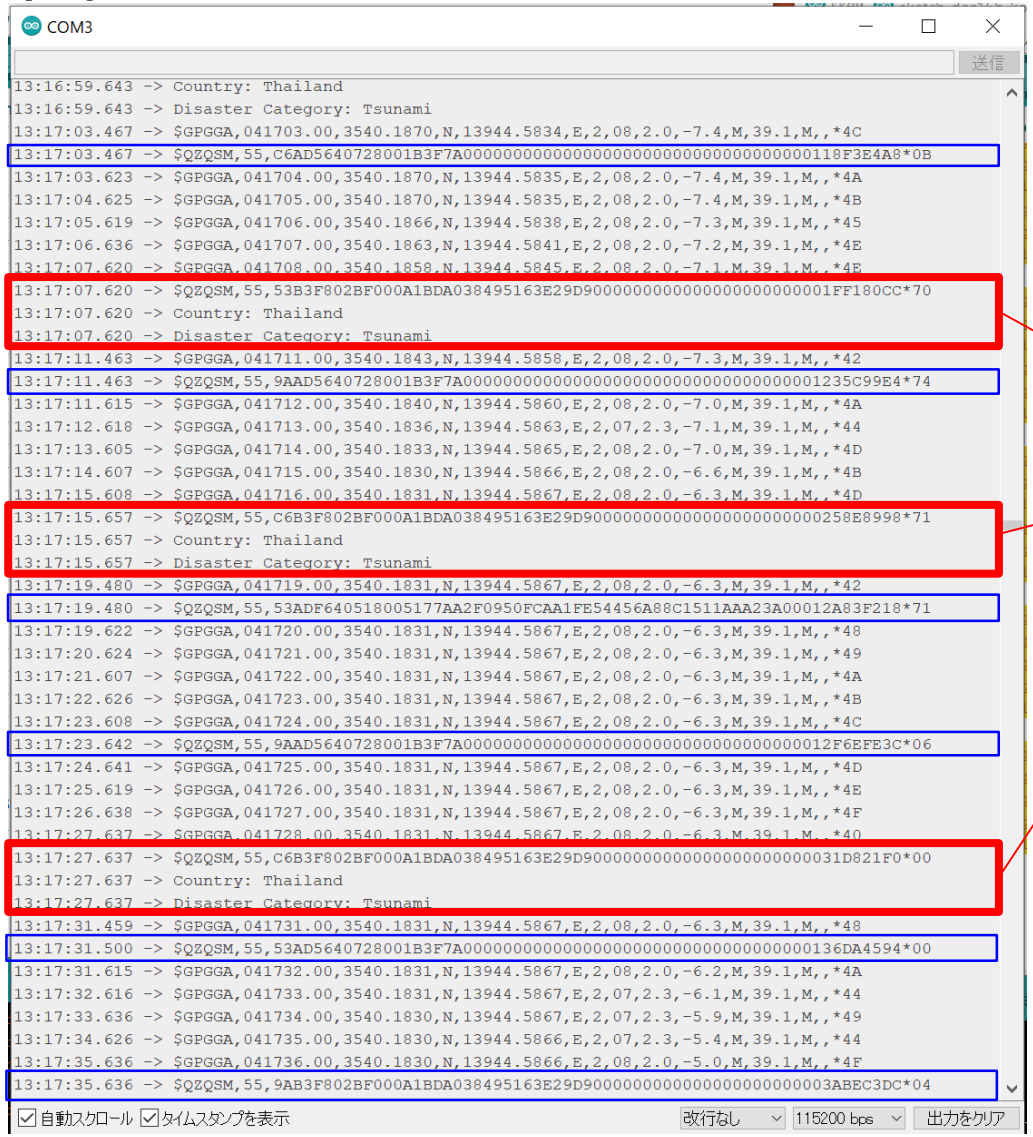
=>011[BIN] = **Met**

Event Sub Category : Bit49 ～ Bit52 (4bits)

=>0001[BIN] = **Flood**

MT44 message data (250bits) to be displayed

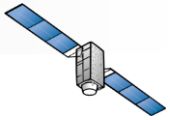
If you can extract the right information from MT44 message, Tsunami Alert will be displayed as below.



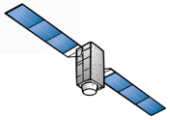
MT44 message

Tsunami will come!!

MT43 message



What is the next step?

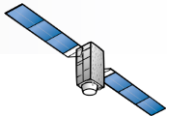


What's the next Step?

(1)Extract current location from positioning result (latitude and longitude)
(2)Extract the target area from EWS message (Latitude and longitude of the center position, Major axis, Minor axis)
(3)Determine whether current location is within target area

For example…
Option 1:Determine with latitude and longitude
- Applying with 1 degree per approx 100km, and determine with latitude and longitude ± 1 degree, by considering a long-angled circle (100km in this time) from the center position.

Option 2:Determine with XY coordinates
- Convert latitude and longitude to XY coordinates. Determine by applying an elliptical formula.



Thank you

Appendix: Sample sketch for mini-Demo on Dec17

```
#include <GNSS.h>
 #include <GNSSPositionData.h>
 #include <gpsutils/cxd56_gnss_nmea.h>
 SpGnss Gnss;
 char PositionData[sizeof(GnssPositionData)];
 /* output NMEA */
 static char nmea_buf[NMEA_SENTENCE_MAX_LEN];

 static char *reqbuf(uint16_t size) {
   if (size > sizeof(nmea_buf)) {
     return NULL;
   }
 return nmea_buf;
 }
 static void freebuf(char *buf) {
   return;
 }
 static int outbin(char *buf, uint32_t len) {
   return len;
 }
 static int outnmea(char *buf) {
   return printf("%s", buf);
 }

 int get_val(const uint8_t *bytes, int startbit, int bitwidth) {
   int val = 0;
   int index = (startbit + bitwidth -1) / 8;
   int lsb = 7 -(startbit + bitwidth -1) % 8;

   int i;
   for (i = 0; i < bitwidth; i++, lsb++) {
     if (lsb > 7) {
     index -= 1;
     lsb = 0;
     }
   val |= ((bytes[index] >> lsb) & 1) << i;
   }
 return val;
 }
```

```
 void setup() {

  /* Initialize Serial */
   Serial.begin(115200);

   /* Initialize GNSS */
   if (Gnss.begin()) {
     Serial.println("begin error!");
   }

   /* select satellite system */
   Gnss.select(GPS); //Gnss.select(GLONASS);
   Gnss.select(QZ_L1CA);
   Gnss.select(QZ_L1S); //Gnss.select(SBAS);

   /* set interval */
   Gnss.setInterval(1);
   if (Gnss.start(COLD_START)) {
     Serial.println("start error!");
   }

   /* use NMEA library */
   NMEA_InitMask();
   NMEA_SetMask(0x4001); // only QZQSM+GGA
   NMEA_OUTPUT_CB funcs;
   funcs.bufReq = reqbuf;
   funcs.out = outnmea;
   funcs.outBin = outbin;
   funcs.bufFree = freebuf;
   NMEA_RegistOutputFunc(&funcs);

 }
```

Appendix: Sample sketch for mini-Demo on Dec17

```
void loop() {
  if (Gnss.waitUpdate(1000)) {
    /* Output NMEA */
    Gnss.getPositionData(PositionData);
    NMEA_Output(&(((GnssPositionData*)PositionData)->Data));
    /* Output QZQSM */
    void *handle;
    if (handle = Gnss.getDCReport()) {
      NMEA_DcReport_Output(handle);

      uint8_t *qzqsm_bytes = ((struct cxd56_gnss_dcreport_data_s*)handle)->sf;
      // For Message Type 44, RPD Challenege EWS trial
      int Coumtry_id = get_val(qzqsm_bytes, 32, 10);
      if (Coumtry_id == 764){
        puts("Country: Thailand");
      }else if(Coumtry_id == 392){
        puts("Country: Japan");
      }
      int Event_category = get_val(qzqsm_bytes, 46, 3);
      int Event_Sub_category = get_val(qzqsm_bytes, 49, 4);
      if (Event_category == 0 && Event_Sub_category == 1){
        puts("Disaster Category: Tsunami");
        for (int j = 0; j < 5; j++){
            digitalWrite(LED0, HIGH);
            digitalWrite(LED3, HIGH);
            delay(100);
            digitalWrite(LED1, HIGH);
            digitalWrite(LED2, HIGH);
            delay(100);

            digitalWrite(LED0, LOW);
            digitalWrite(LED3, LOW);
            delay(10);
            digitalWrite(LED1, LOW);
            digitalWrite(LED2, LOW);
            delay(10);

            digitalWrite(LED0, HIGH);
            digitalWrite(LED1, HIGH);
            delay(100);
            digitalWrite(LED2, HIGH);
            digitalWrite(LED3, HIGH);
            delay(100);

            digitalWrite(LED0, LOW);
            digitalWrite(LED1, LOW);
            delay(10);
            digitalWrite(LED2, LOW);
            digitalWrite(LED3, LOW);
            delay(10);

            digitalWrite(LED2, HIGH);
            digitalWrite(LED3, HIGH);
            delay(100);
            digitalWrite(LED0, HIGH);
            digitalWrite(LED1, HIGH);
            delay(100);

            digitalWrite(LED2, LOW);
            digitalWrite(LED3, LOW);
            delay(10);
            digitalWrite(LED0, LOW);
            digitalWrite(LED1, LOW);
            delay(10);

            digitalWrite(LED0, HIGH);
            digitalWrite(LED3, HIGH);
            digitalWrite(LED1, HIGH);
            digitalWrite(LED2, HIGH);
            delay(100);

            digitalWrite(LED0, LOW);
            digitalWrite(LED3, LOW);
            digitalWrite(LED1, LOW);
            digitalWrite(LED2, LOW);
            delay(10);
    }
  }
```

Appendix: Sample sketch for mini-Demo on Dec17

```
else if(Event_category == 3 && Event_Sub_category == 1){
  puts("Disaster Category: Flood");
  for (int j = 0; j < 5; j++){
    digitalWrite(LED0, HIGH);
    digitalWrite(LED3, HIGH);
    delay(100);
    digitalWrite(LED1, HIGH);
    digitalWrite(LED2, HIGH);
    delay(100);

    digitalWrite(LED0, LOW);
    digitalWrite(LED3, LOW);
    delay(10);
    digitalWrite(LED1, LOW);
    digitalWrite(LED2, LOW);
    delay(10);

    digitalWrite(LED0, HIGH);
    digitalWrite(LED1, HIGH);
    delay(100);
    digitalWrite(LED2, HIGH);
    digitalWrite(LED3, HIGH);
    delay(100);

    digitalWrite(LED0, LOW);
    digitalWrite(LED1, LOW);
    delay(10);
    digitalWrite(LED2, LOW);
    digitalWrite(LED3, LOW);
    delay(10);

    digitalWrite(LED2, HIGH);
    digitalWrite(LED3, HIGH);
    delay(100);
    digitalWrite(LED0, HIGH);
    digitalWrite(LED1, HIGH);
    delay(100);
```

```
    digitalWrite(LED2, LOW);
    digitalWrite(LED3, LOW);
    delay(10);
    digitalWrite(LED0, LOW);
    digitalWrite(LED1, LOW);
    delay(10);

    digitalWrite(LED0, HIGH);
    digitalWrite(LED3, HIGH);
    digitalWrite(LED1, HIGH);
    digitalWrite(LED2, HIGH);
    delay(100);

    digitalWrite(LED0, LOW);
    digitalWrite(LED3, LOW);
    digitalWrite(LED1, LOW);
    digitalWrite(LED2, LOW);
    delay(10);
  }
 }
}
}
}
```