Experiments on Communication Networks_ Week5

2022-2학기

YONSEI UNIVERSITY

ralio

Overview of Experiments

WEEK1 – Python Visualization

WEEK1 – LABVIEW tutorial

WEEK1 – Channel sensor tutorial

WEEK2 – Active sensing

WEEK2 – Radar generator tutorial

WEEK2 – Making datasets

WEEK3 – Data labeling

WEEK3 – Training/ Testing CNN model

Week 2 – Detail Contents

1. Active sensing
2. US CBRS & SAS
3. Yonsei CBRS testbed
4. Radar generator tutorial
5. Spectrum sensor tutorial
6. Making datasets

2주차 실습 개요

- USRP 실습은 다음의 과정을 통해 이루어집니다.
  - **1주차**
    - Labview 설치 및 radar generator 코드 간단한 설명 및 실행.
    - Python 설치 및 IQdata 를 spectrogram으로 visualization 실행.
  - **2주차**
    - Active sensing 에 대한 이해.
    - 미국 CBRS 대역의 SAS 시스템과 이 기반의 연세 testbed 소개.
    - Radar generator 코드 상세 설명.
    - Spectrum sensor 실행 방법 소개.
    - Radar generator 와 spectrum sensor 을 동시 실행하여 만들어진 IQ data를 spectrogram 으로 visualization 하여 데이터셋 생성.
  - **2주차 시작하기 전에 앞서**
    - 2주차의 내용은 1주차에 소개드린 내용에 기반하여 좀 더 심화된 단계입니다.
    - 필요할 시 1주차에 진행한 매뉴얼들을 참고하시면 좋습니다.

Remote Sensing

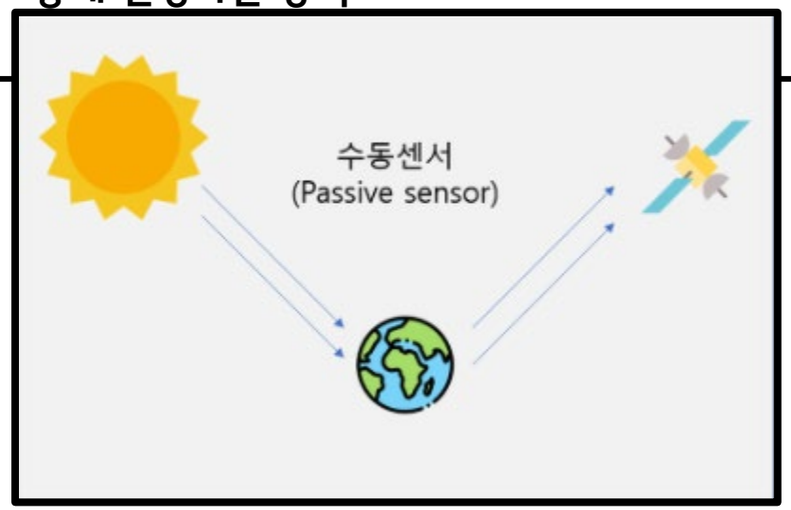
Remote Sensing

- 측정하고자 하는 목표물을 직접 측정하지 않고, 목표물에서 반사 또는 복사되어 나오는 전자기파를 감지하여 그 물리적 성질을 측정하는 기술
- Remote sensing 을 통해 측정하는 과정 중, 정보 습득하는 방법으로 두가지 방식으로 나뉨

  1) Passive sensing 2) Active Sensing
- 어떤 종류의 센서를 사용하는지 유무에 따라 두 가지 방식으로 나뉨 : 1) passive sensor 2) active sensor

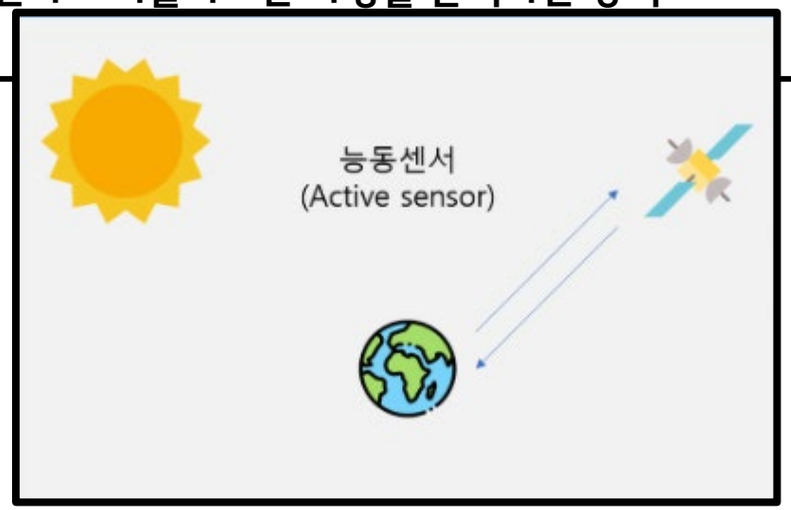
Passive Sensor

- 태양 에너지를 통해 반사되거나 복사되어 나오는 정보를 수신 받아 정보를 얻는 방식
- Sensor 가 Sensor 이외의 에너지원으로부터 에너지를 통해 센싱하는 방식



Active Sensor

- 인공적으로 만들어진 전자기 에너지를 직접 쏘아 센서로 되돌아오는 복사 에너지를 분석하는 방법
- Sensor 가 직접 에너지를 쏘아 해당 에너지파로 부터 센서로 되돌아오는 파형을 분석하는 방식



수동센서

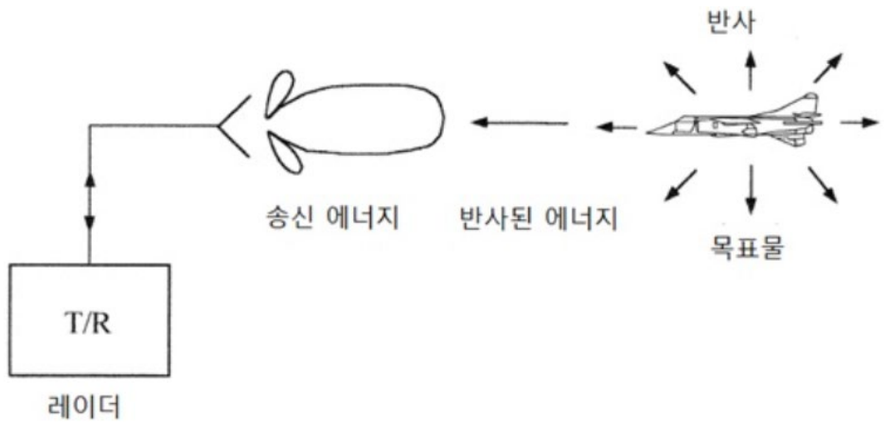
- 카메라, 전자광학 센서, 적외선 센서

능동센서

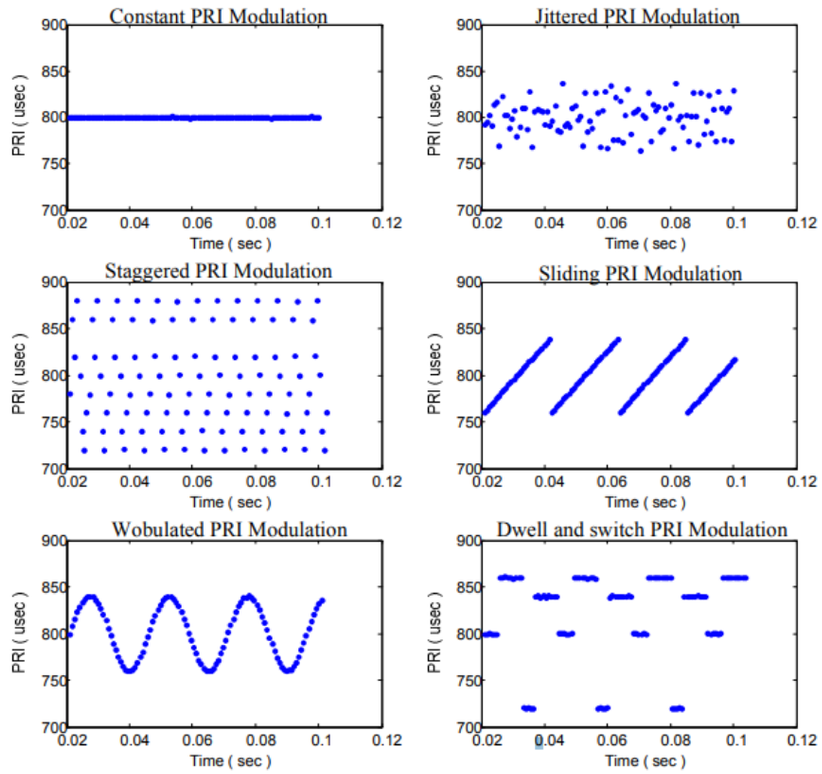
- 전자기파, 레이더

Radar signal

- Radar (**RA**dio **D**etection **A**nd **R**anging)
  - **전파를 사용하여 목표물의 거리, 방향, 각도 및 속도를 측정하는 감지 시스템**
  - **전자파를 방사해서 측정하고자 하는 목표물에 맞고 반사되어 되돌아 오늘 전자파를 분석하여 대상물과의 거리를 측정하는 원리**
  - **목표물까지의 거리: 송신된 신호가 다시 되돌아오는 시간으로부터 계산**
  - **목표물의 상대적인 속도: 되돌아 온 신호의 도플러 변이로부터 계산**

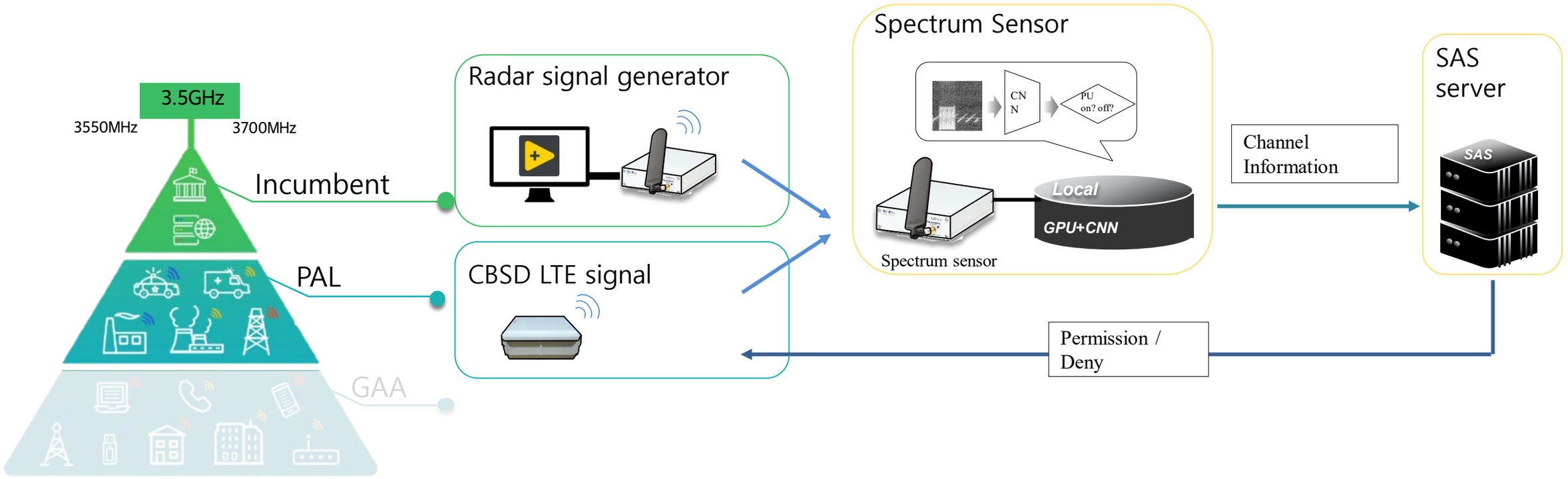


- Common modulation types
  - **Remote sensing 을 하기 위해 여러가지 파형의 radar 를 사용**
  - **가장 흔히 사용하는 radar 파의 종류: Pulsed radar**
  - **Pulsed radar 는 pulse duration T 와 pulse repetition interval(PRI) 로 특징 지을 수 있다.**
  - **Pulsed radar 는 다양한 pulse duration 과 PRI 를 갖도록 각각 다른 패턴의 파형을 사용**
  - **오른쪽 그림은 실제 해안에서 관측되는 여러가지 형태의 레이더 신호**



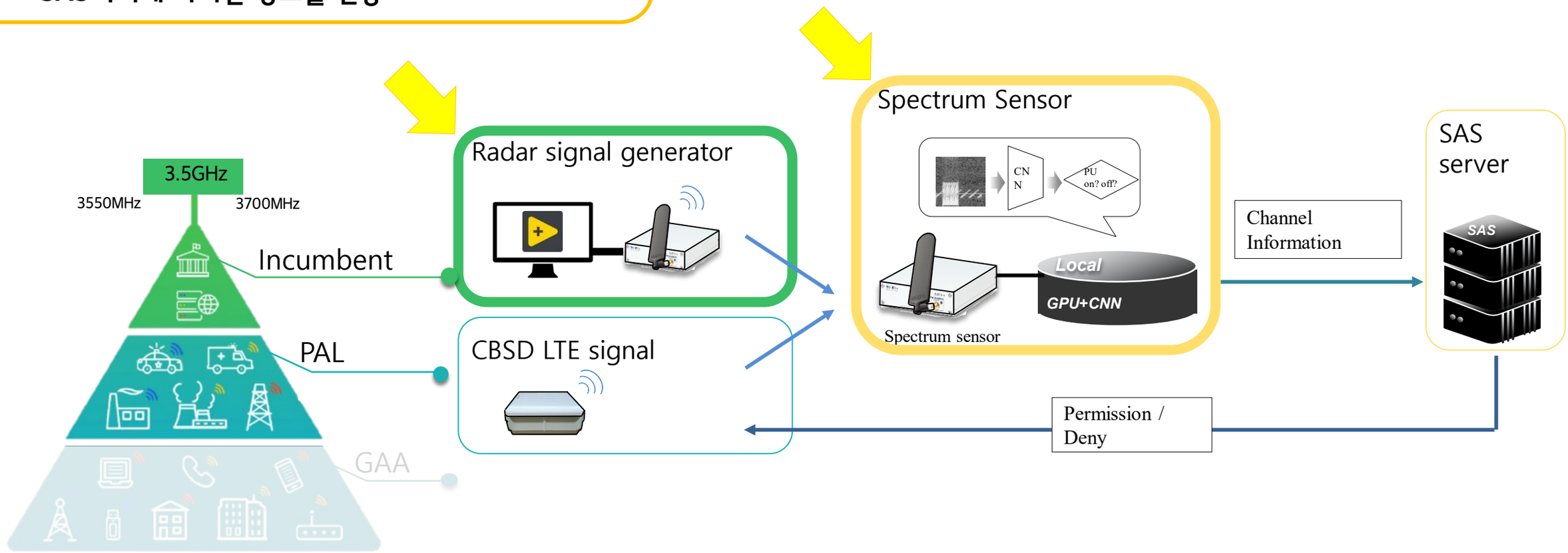
Yonsei CBRS testbed

- Radar signal generator
  - **1차 사용자. 연방, 위성 시스템 신호를 송출**
- CBSD LTE signal
  - **2차 사용자. LTE 신호를 송신**
- Spectrum Sensor
  - **3.5GHz 대역에 보내지고 있는 신호를 수신하여 1,2차 사용자의 사용 여부를 감지,**
  - **SAS서버에 이러한 정보를 전송**
- SAS server
  - **2차 사용자에 주파수 사용 (신호 송신) 권한을 실시간으로 부여**
  - **Ex) 1차 사용자와 동시에 사용할 경우 주파수 사용 금지 권고**



Yonsei CBRS testbed

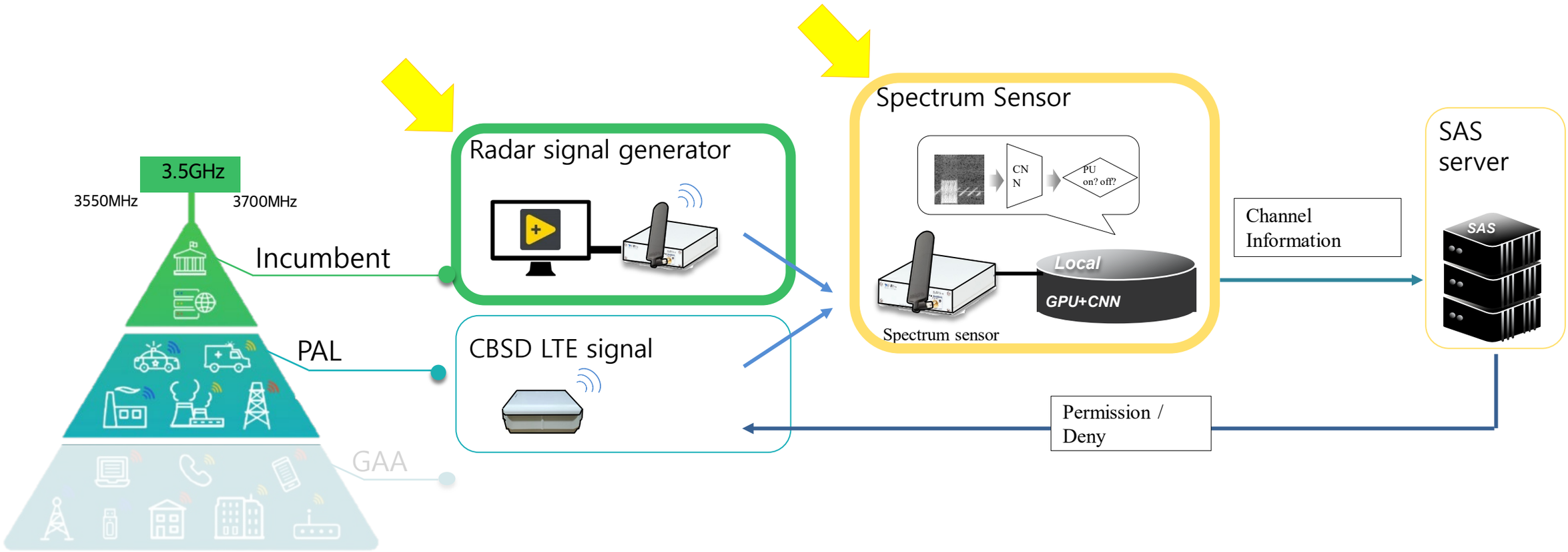
- Radar signal generator
  - **1차 사용자. 연방, 위성 시스템 신호를 송출**
- CBSD LTE signal
  - **2차 사용자. LTE 신호를 송신**
- Spectrum Sensor
  - **3.5GHz 대역에 보내지고 있는 신호를 수신하여 1,2차 사용자의 사용 여부를 감지,**
  - **SAS서버에 이러한 정보를 전송**
- SAS server
  - **2차 사용자에 주파수 사용 (신호 송신) 권한을 실시간으로 부여**
  - **Ex) 1차 사용자와 동시에 사용할 경우 주파수 사용 금지 권고**



Yonsei CBRS testbed

- 본 실험에서는 Radar signal generator 과 Spectrum Sensor 를 구현할 예정입니다
  - **Week1 에서 소개한 Radar signal generator LABVIEW 코드를 더 자세히 알아보고 다양한 스펙의 radar 파형을 구현한다**
  - **Spectrum sensor 는 Radar signal generator 가 송출한 Radar 를 센싱하여**

    해당 주파수 채널에 radar 가 있는지를 감지한다
  - **Spectrum Sensor 는 이를 위해 수신한 신호 파형을 이미지 형태로 변환해야한다**
  - **이미지 형태의 신호를 딥러닝을 통해 해당 주파수 채널에 radar가 있는지 감지한다**

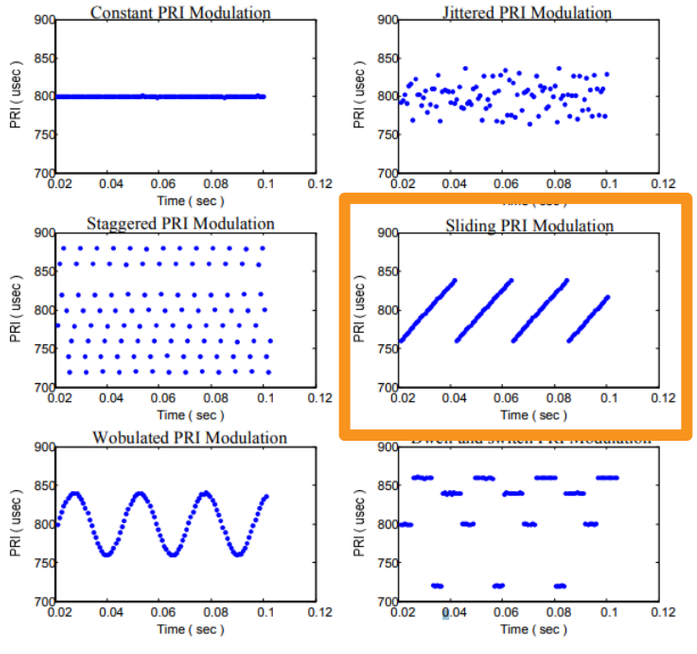


Radar Signal Generator

Common modulation types

- **이 중 Linear frequency modulation,**

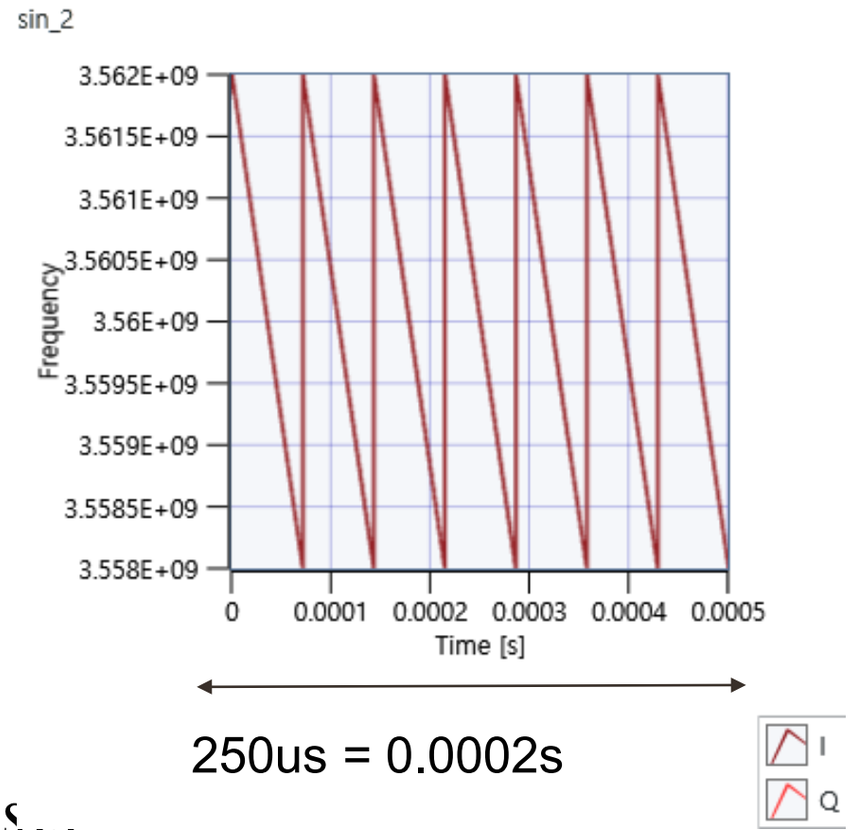
  즉 chirp 파형을 생성하는 generator을 구현할 예정



Spectrogram

- **시간-주파수 축**

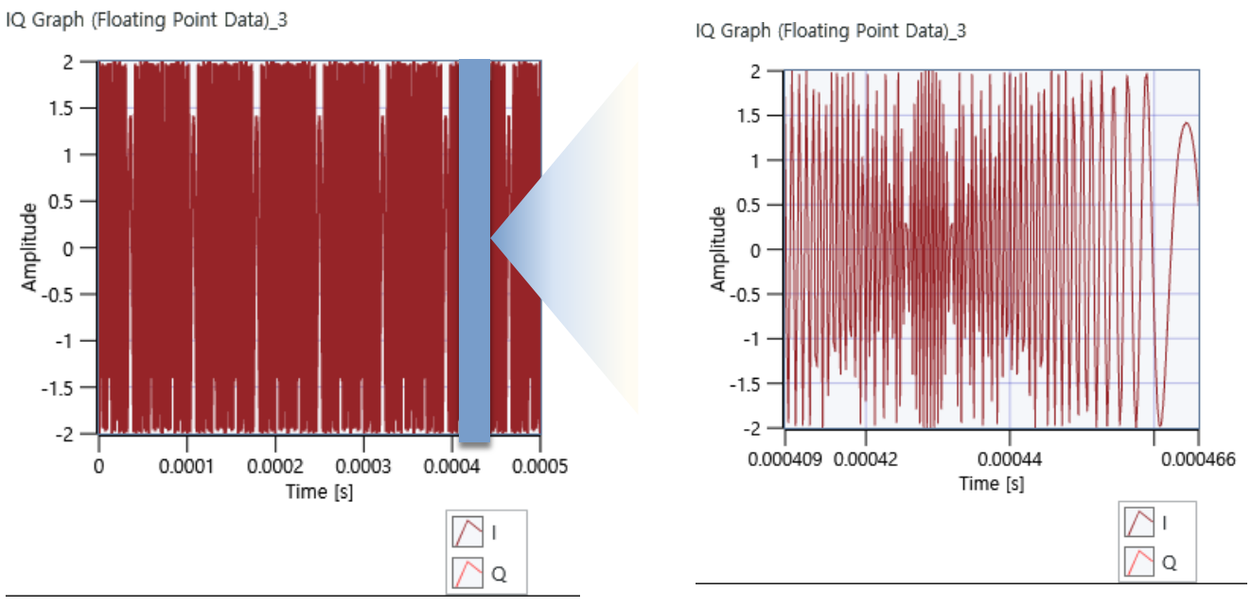
20MHz



250us = 0.0002s

IQ signal

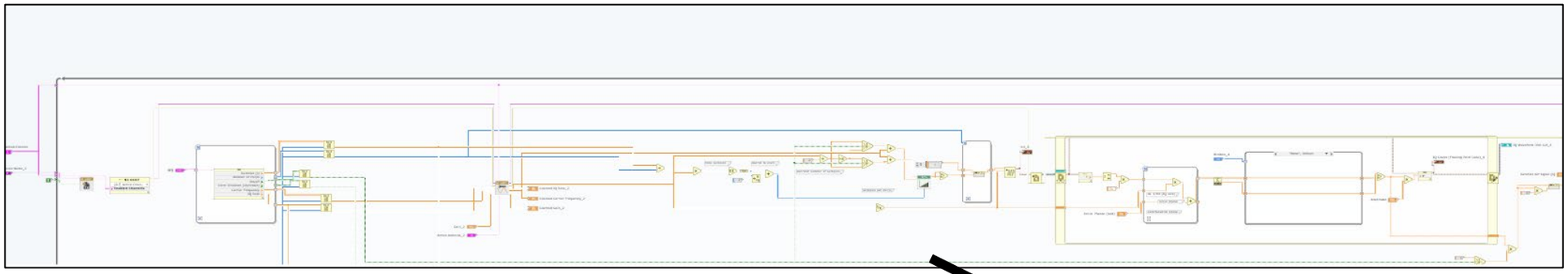
- **시간- Amplitude 축**
- **실제 USRP 안테나가 전송하는 신호**



코드 소개

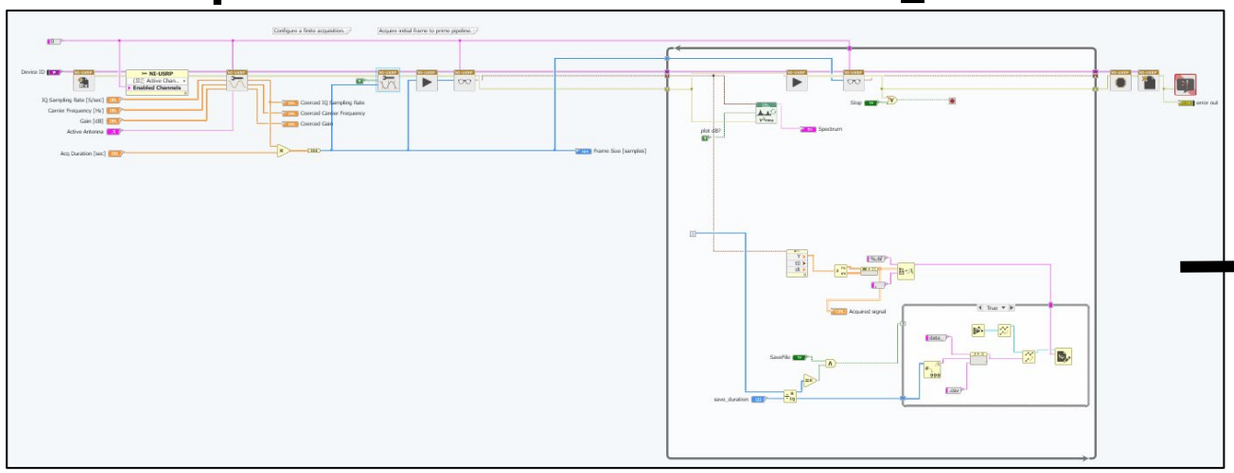
❖ 다이어그램

▪ Radar signal generator: Radar_TX



송신: TX2900

▪ Spectrum sensor: Radar_RX



수신: NI2900

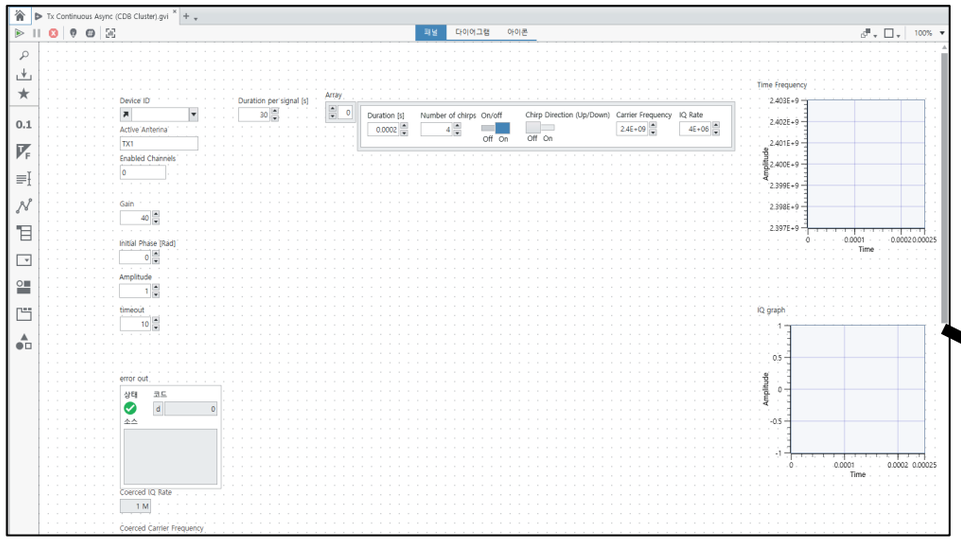
```python
import torch
import numpy as np
import torch.nn as nn
import torch.nn.functional as F
from torch.utils.data import Dataset, DataLoader, random_split
import numpy as np
import pandas as pd
import os
from datetime import datetime
import matplotlib.pyplot as plt
import sys
import cv2

def main():

    data_root = "./IQ_signal"
    write_root = "./spectrogram"

    MyDataSet = CustomDataSet(data_root)
    data_length = MyDataSet.__len__()
    for i in range(data_length):
        [datum, target]= MyDataSet.__getitem__(i)
        name = target[0]
        cv2.imwrite(write_root +'/' + name[0:-4] +'.png', datum[0])
```

Python Visualization

코드 소개

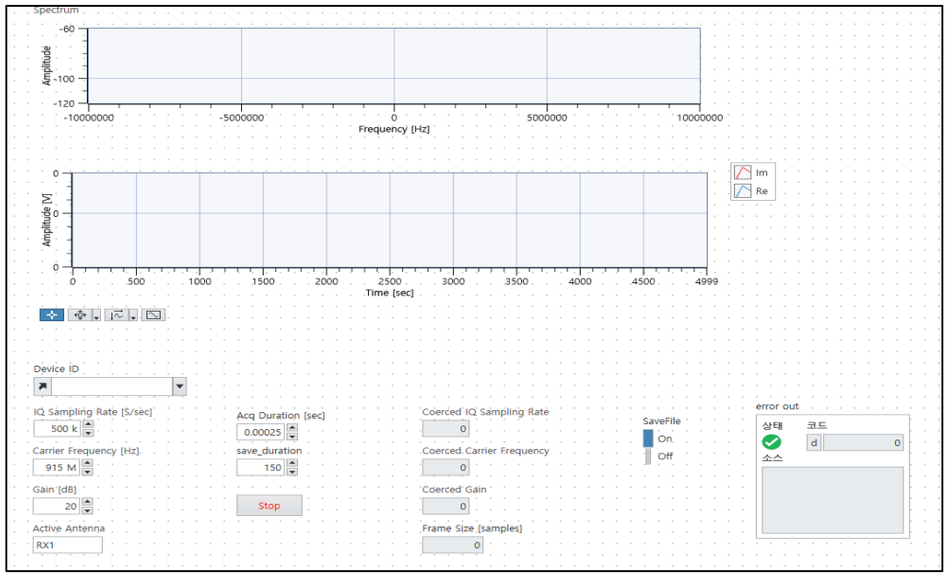
❖ 패널

- **Radar signal generator: Radar_TX**



송신: TX2900

- **Spectrum sensor: Radar_RX**



수신: NI2900

```python
import torch
import numpy as np
import torch.nn as nn
import torch.nn.functional as F
from torch.utils.data import Dataset, DataLoader, random_split
import numpy as np
import pandas as pd
import os
from datetime import datetime
import matplotlib.pyplot as plt
import sys
import cv2

def main():

    data_root = "./IQ_signal"
    write_root = "./spectrogram"

    MyDataSet = CustomDataSet(data_root)
    data_length = MyDataSet.__len__()
    for i in range(data_length):
        [datum, target]= MyDataSet.__getitem__(i)
        name = target[0]
        cv2.imwrite(write_root +'/' + name[0:-4] +'.png', datum[0])
```

Python Visualization

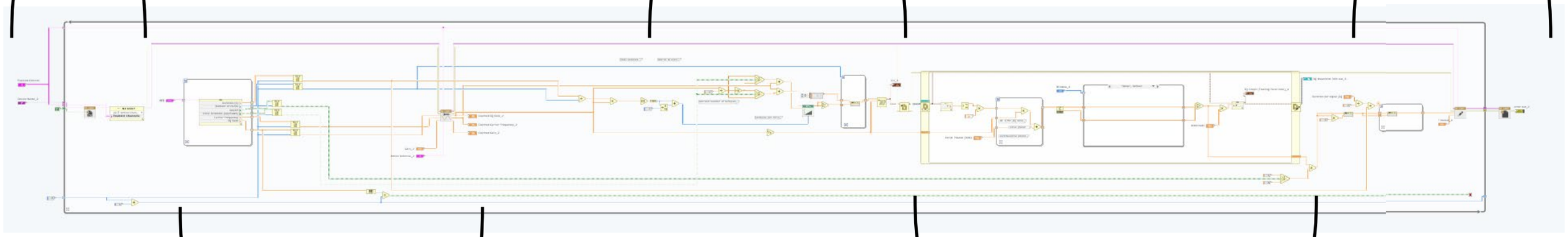
송신 파트: CBRS_chirp_TX

❖ 전체 코드의 흐름

USRP와 연결

시간-주파수 축의
Chirp 신호파형 생성

생성한 신호를 Duration동안
반복하여 USRP로 송출



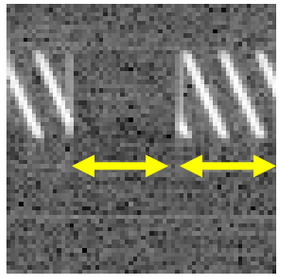
입력한 인풋값 불러오기

생성한 파형을 시간-Amplitude 축의
IQ signal 로 변환

송신) 입력값

❖ Duration

- Periodic 한 chirp sample을 생성함



❖ Number of chirps

- 한 Duration 안에 생성되는 Chirp신호 (사선) 개수
- Ex) 7

❖ On/off

- 신호 생성 여부
- On 으로 항상 설정하면 됨

❖ Chirp Direction

- 대각선의 방향

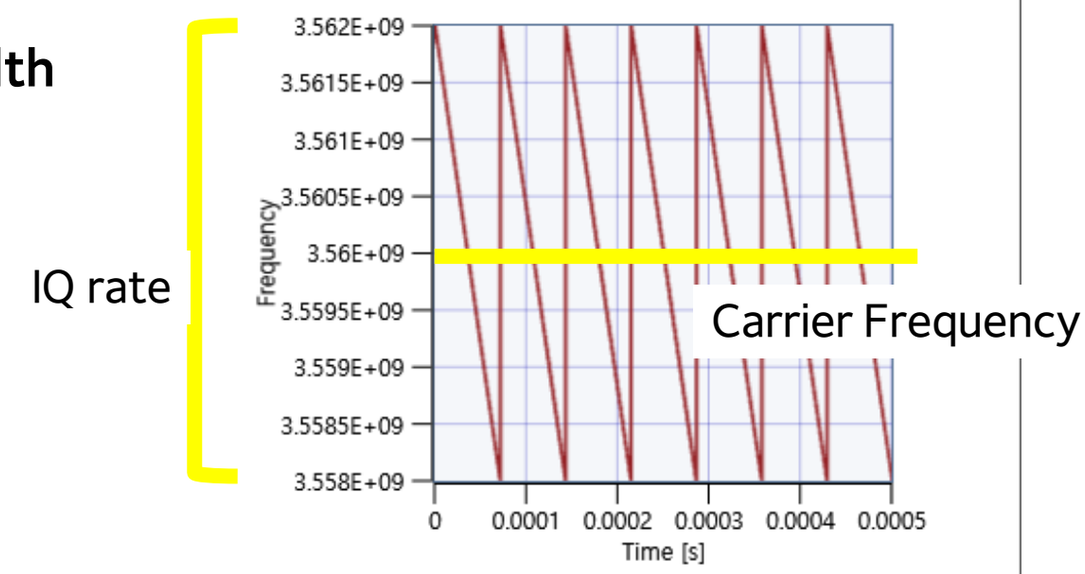
❖ Carrier Frequency

- 신호의 중심주파수
- Ex) 3.56G (실제 실행은 2.39-2.41G 사이 값으로 해야함)

❖ IQ rate

- Bandwidth
- Ex) 4M

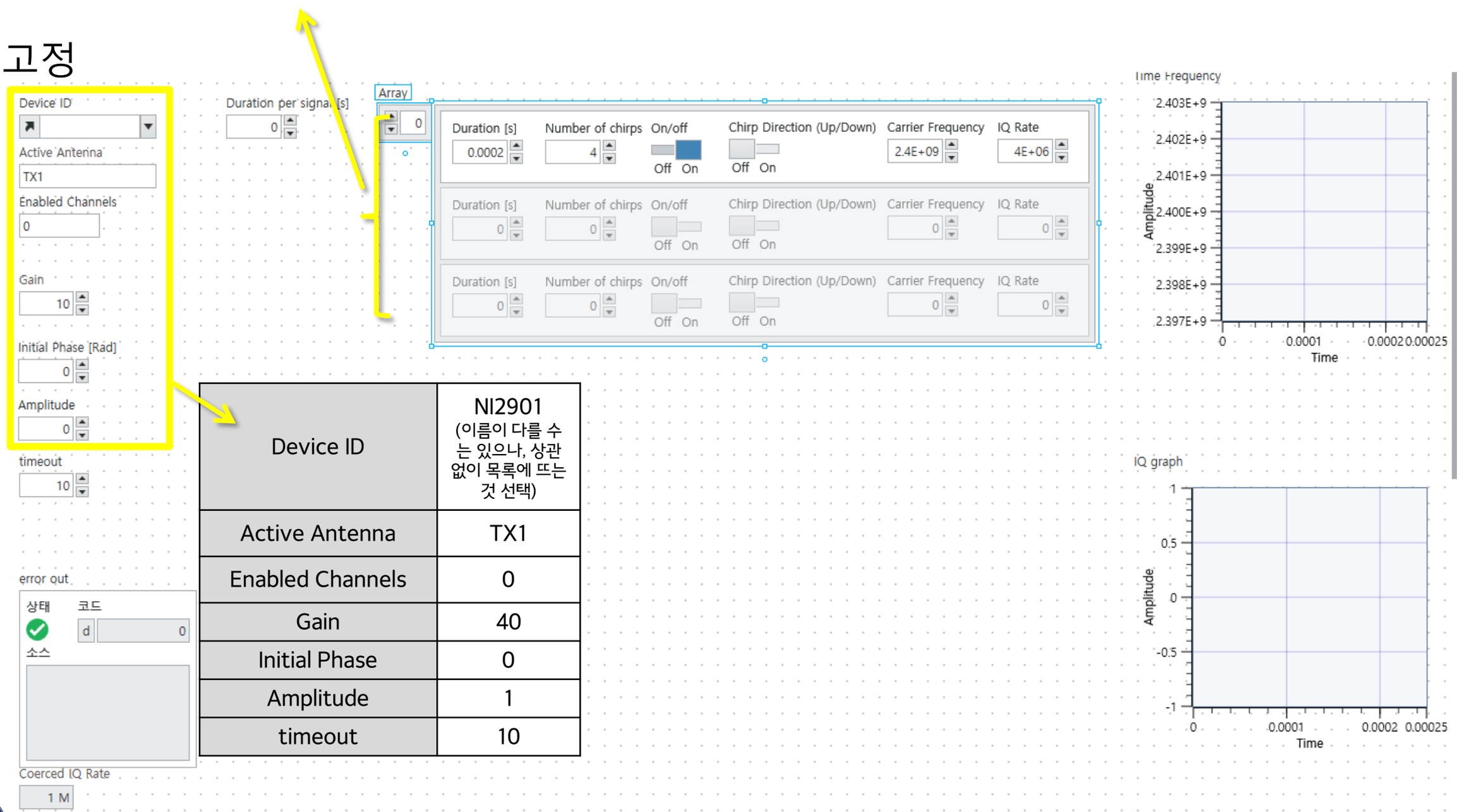
Ex



송신) 입력값 (패널)

- 여러 인풋값을 한번에 입력하고, 한번 실행에 모든 인풋값에 대한 코드가 돌아가도록 한다
- 인풋값들의 '배열' 이라 하고 배열 순서대로 인풋값이 코드에 적용되어 실행된다

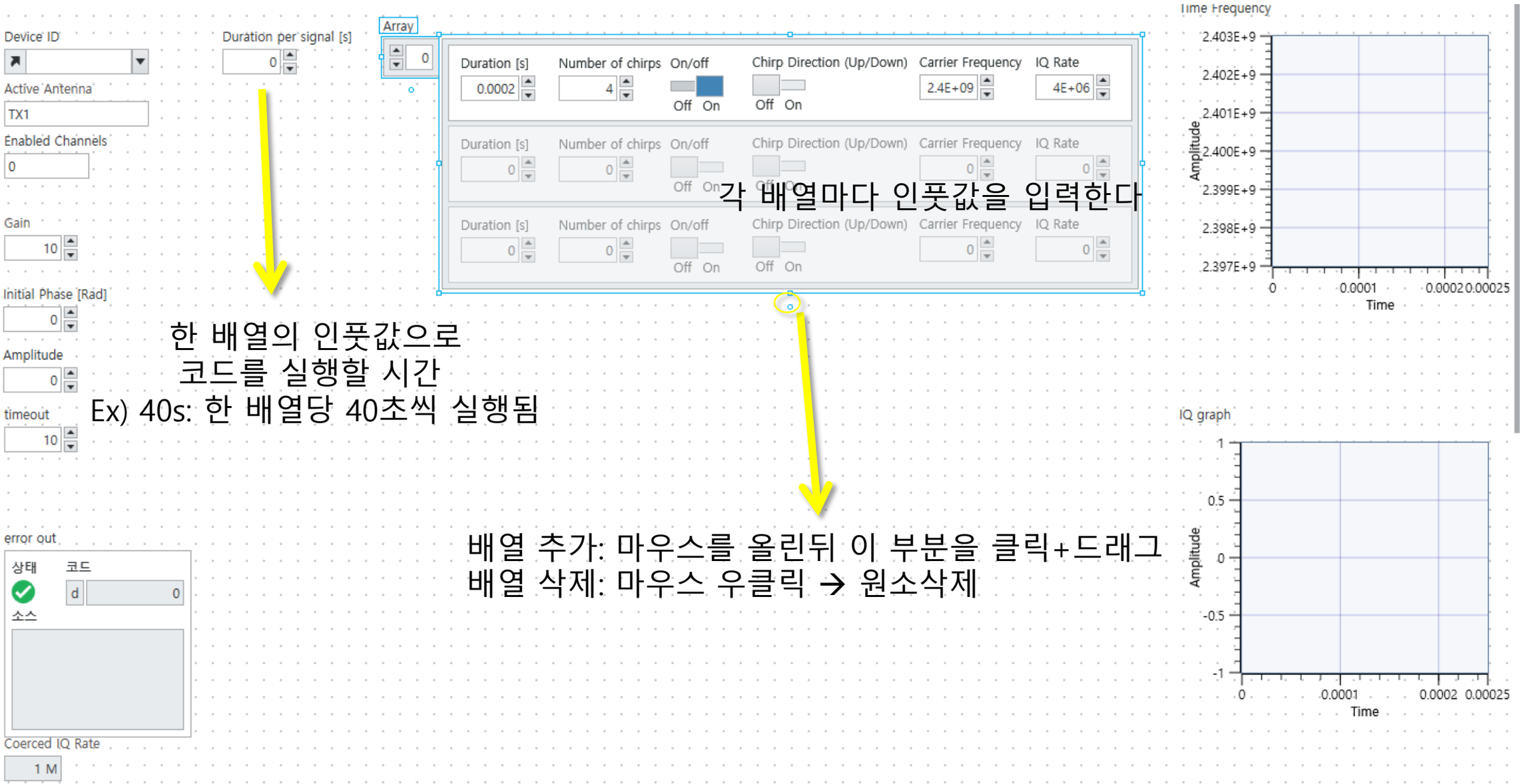
고정



Device ID	NI2901 (이름이 다를 수는 있으나, 상관없이 목록에 뜨는 것 선택)
Active Antenna	TX1
Enabled Channels	0
Gain	40
Initial Phase	0
Amplitude	1
timeout	10

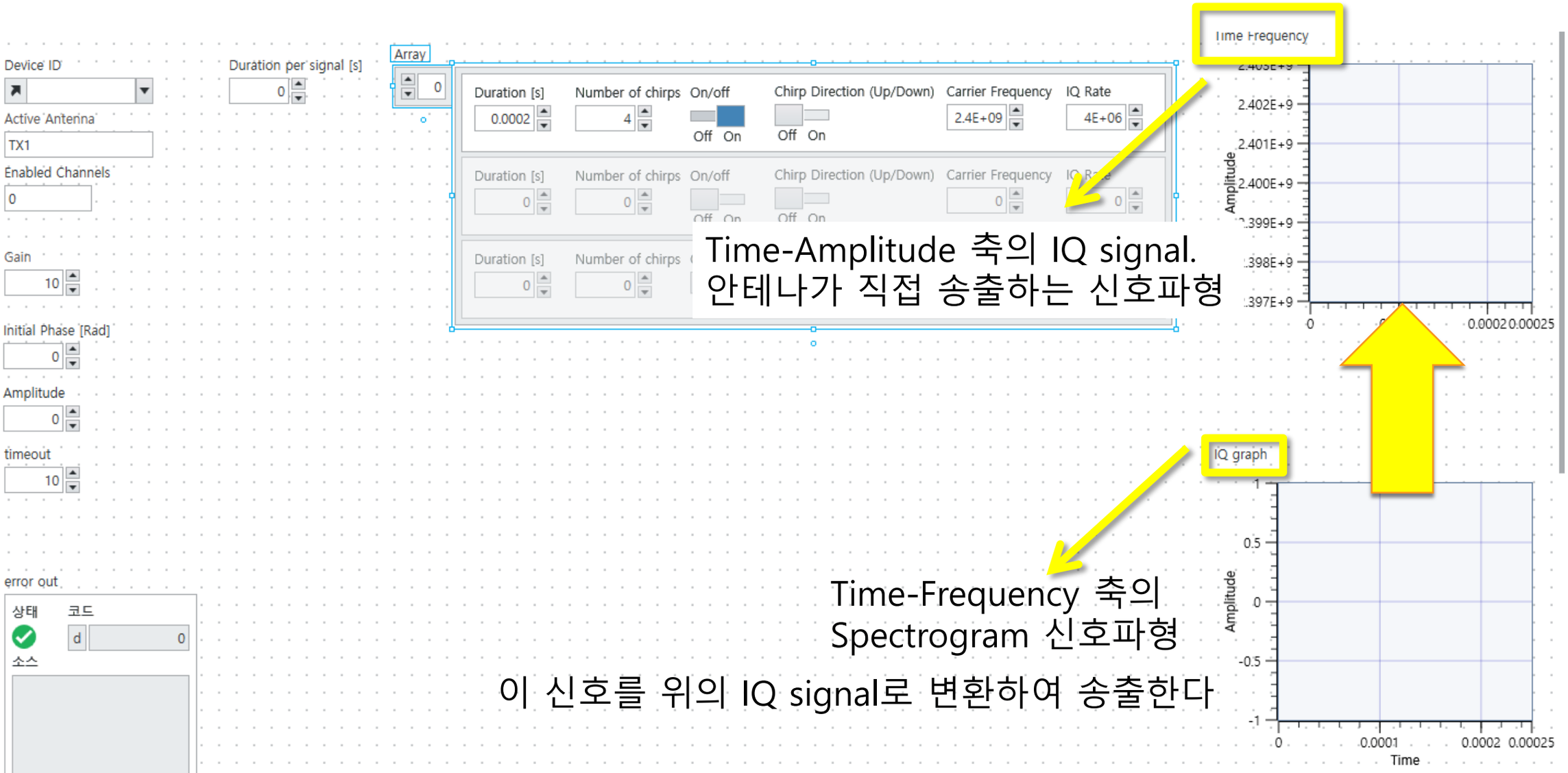
송신) 입력값 (패널)

- 여러 인풋값을 한번에 입력하고, 한번 실행에 모든 인풋값에 대한 코드가 돌아가도록 한다
- 인풋값들의 '배열' 이라 하고 배열 순서대로 인풋값이 코드에 적용되어 실행된다



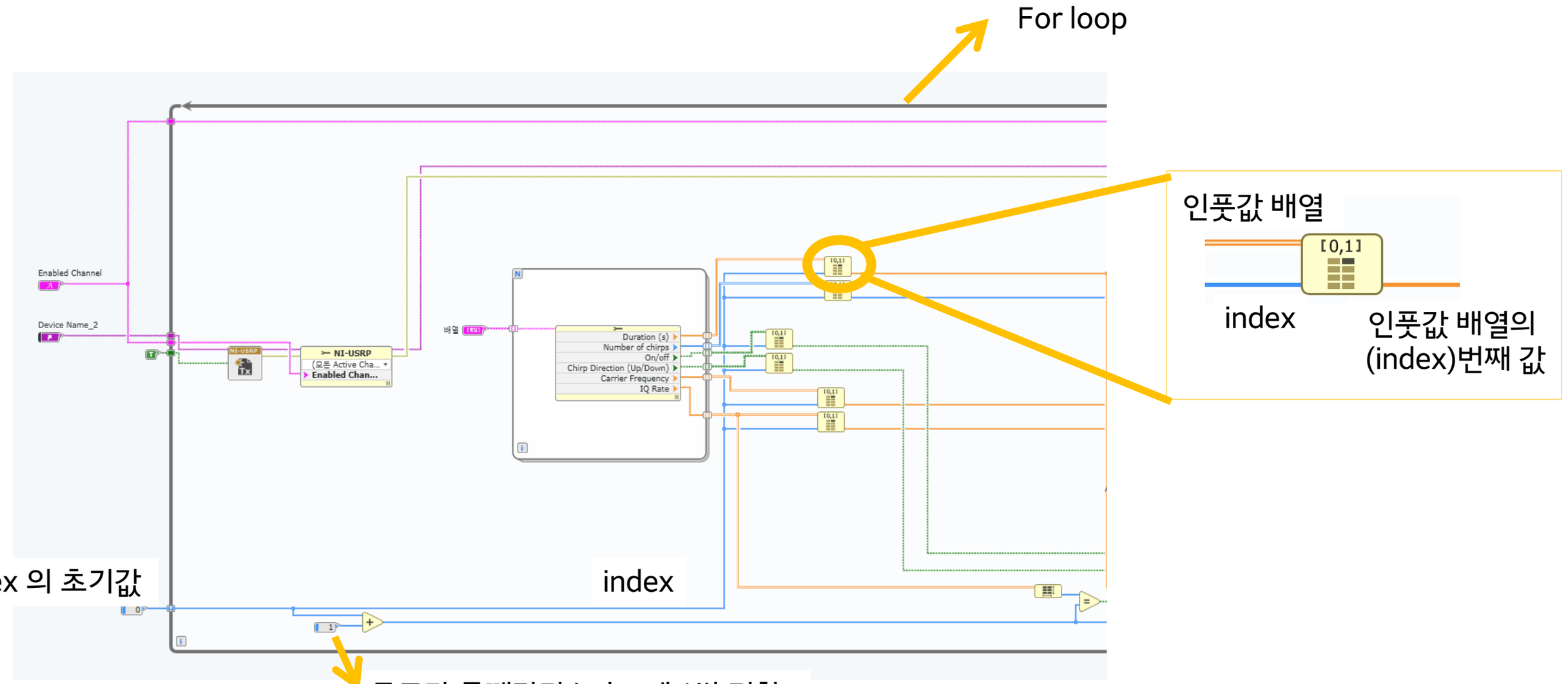
송신) 출력값 (패널)

- 여러 인풋값을 한번에 입력하고, 한번 실행에 모든 인풋값에 대한 코드가 돌아가도록 한다
- 인풋값들의 '배열' 이라 하고 배열 순서대로 인풋값이 코드에 적용되어 실행된다



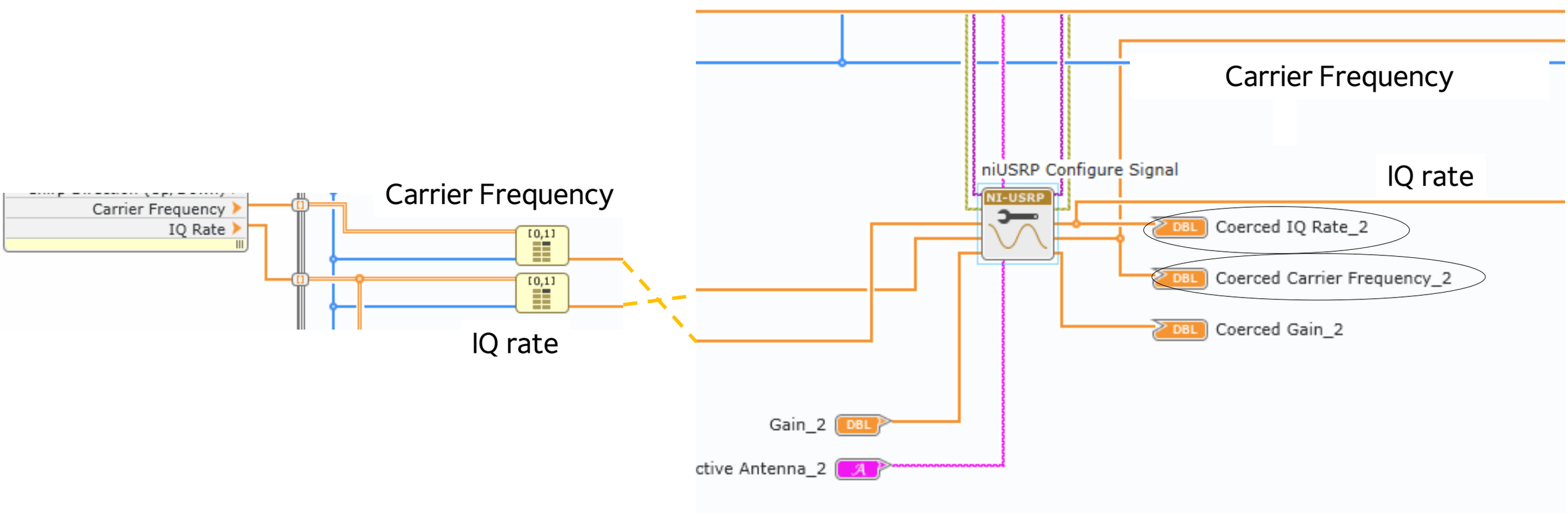
송신) 입력값 (다이어그램)

- 패널에 입력한 인풋값의 배열을 하나씩 꺼내어 사용
- 전체 코드를 while문 안에 넣어 한 루프가 돌때마다 index +1 취하고,
- 인풋값 배열에서 (index)번째의 값을 꺼내 사용한다



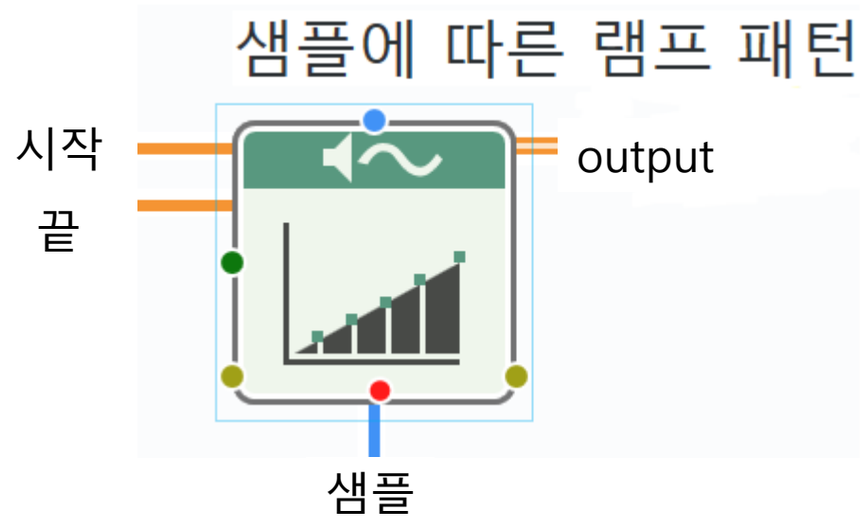
송신) 입력값 (다이어그램)

- 입력값 중 Carrier frequency, IQ rate 값은 ‘niUSRP Configure Signal’ 을 거쳐야 한다
- 배열에서 나온 Carrier frequency, IQ rate 값을 ‘niUSRP Configure Signal’ 의 해당 부분에 입력한다
- ‘niUSRP Configure Signal’ 의 출력으로 나온 값을 사용한다



송신) Chirp 생성 블록

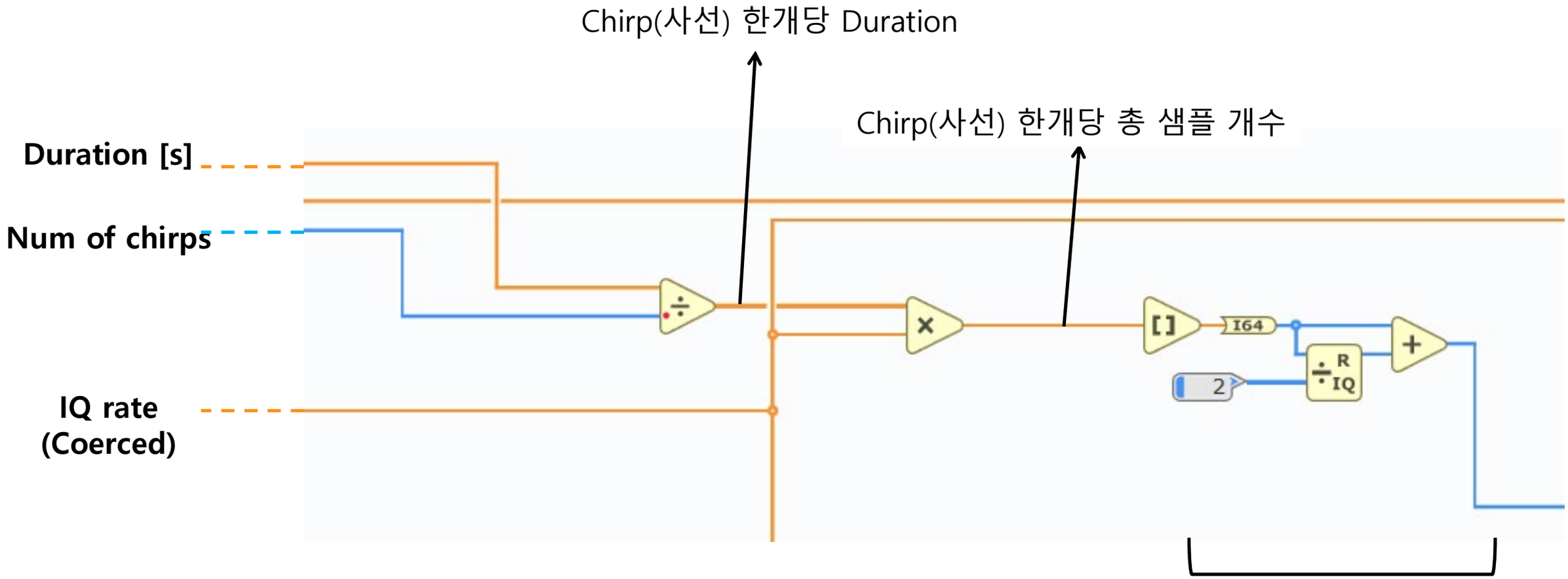
- 하나의 사선(Chirp) 신호를 만들어주는 블록
  - **시작: 사선의 시작값**
  - **끝: 사선의 마지막 값**
  - **샘플: 한 사선을 이루는 샘플 개수**
  - **output: 생성된 사선 신호 데이터**



- 기본 도움말 및 온라인 매뉴얼 참고

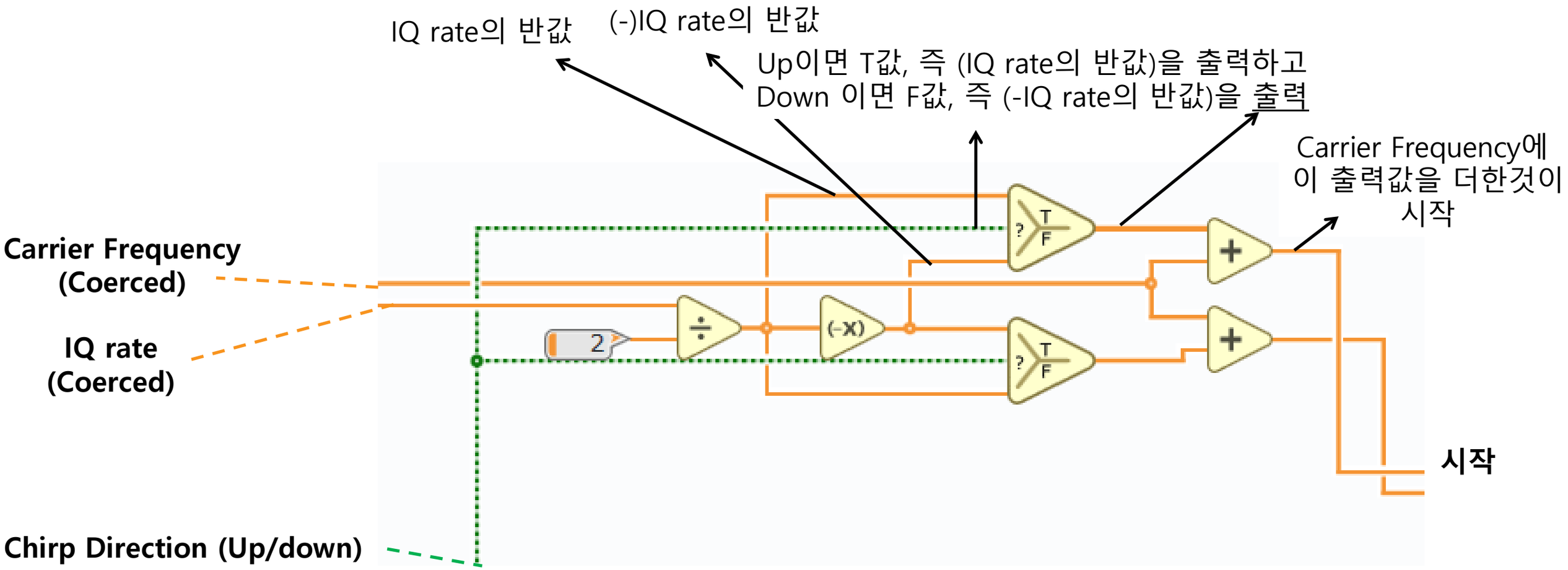
송신) chirp 블록 샘플

- Bandwidth (IQ rate)가 클수록 선명하며, 수신할 때 decoding 이 잘 되는 신호를 만든다

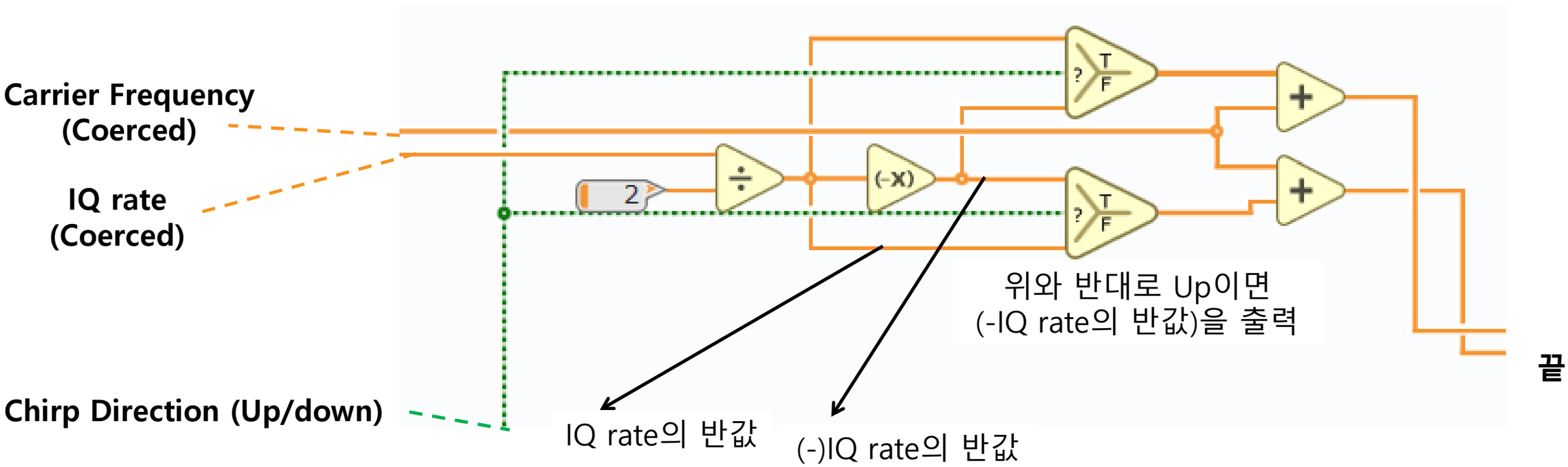


  - 각 블록의 기본 도움말 및 온라인 매뉴얼 참고

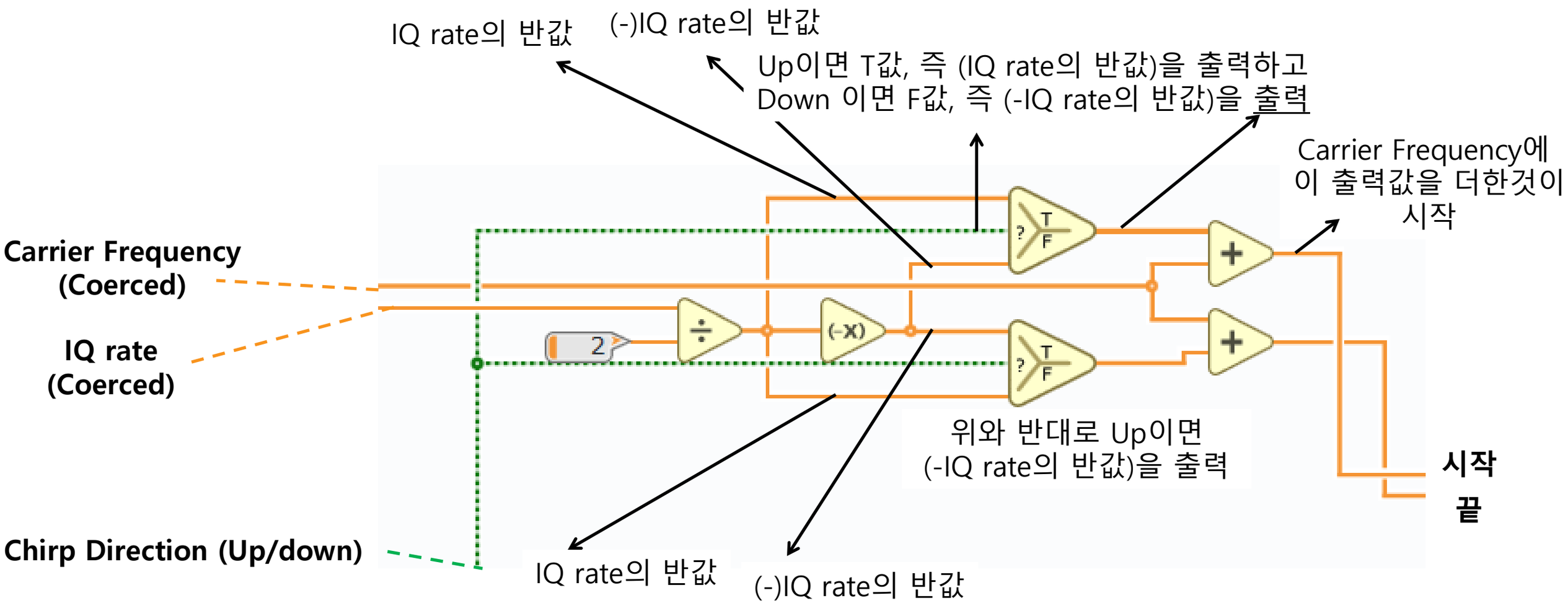
송신) chirp 블록 시작, 끝



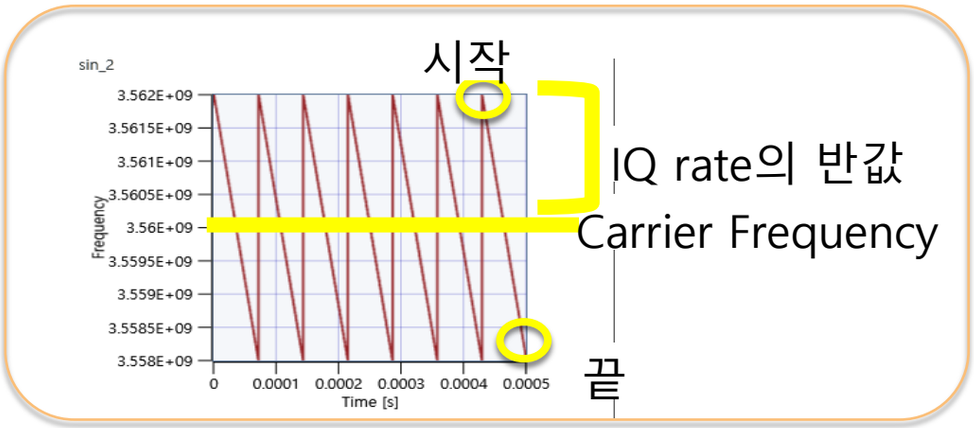
송신) chirp 블록 시작, 끝



송신) chirp 블록 시작, 끝

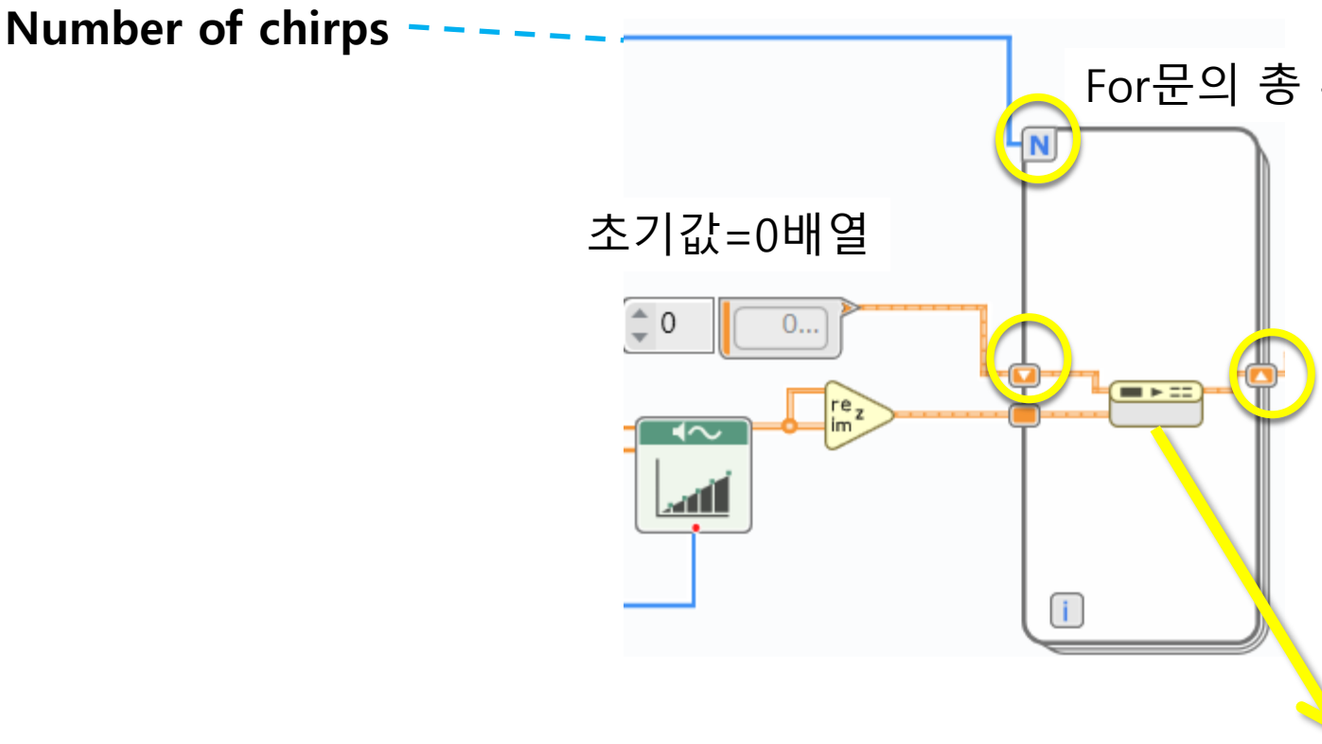


- Chirp Direction이 Up 일 경우:
  - **시작: IQ rate의 반값 + Carrier frequency**
  - **끝: -IQ rate의 반값 + Carrier fruquency**



송신) 데이터 생성

- 생성된 chirp 신호 하나의 데이터를 Number of chirps 만큼 반복한다

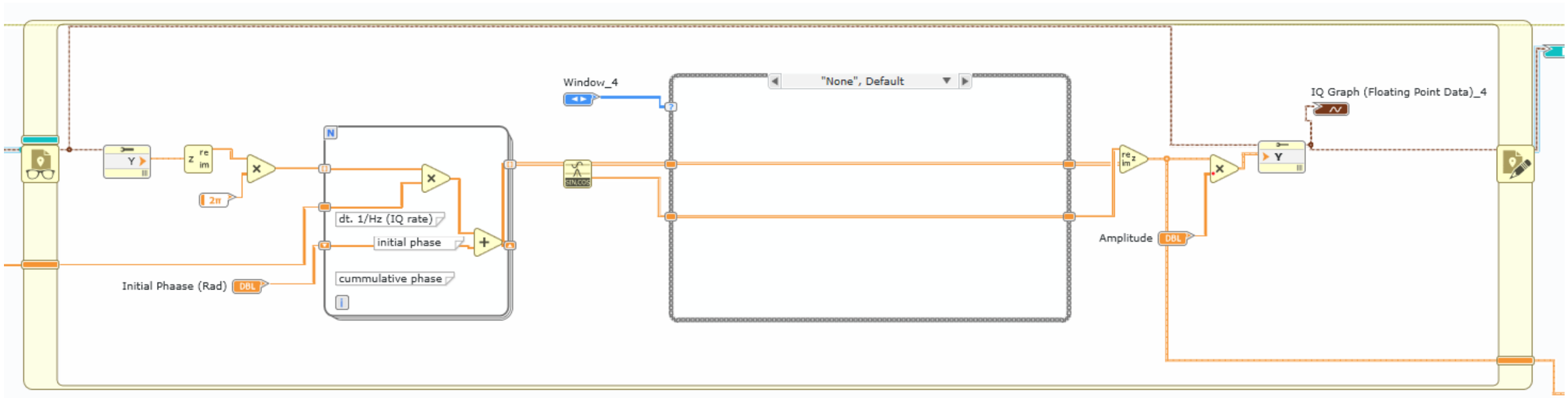


For문에 다음과 같은 화살표 모양의 터널은 출력값(오른쪽)을 다시 입력으로(왼쪽) 넣어 루프를 실행한다는 뜻이다

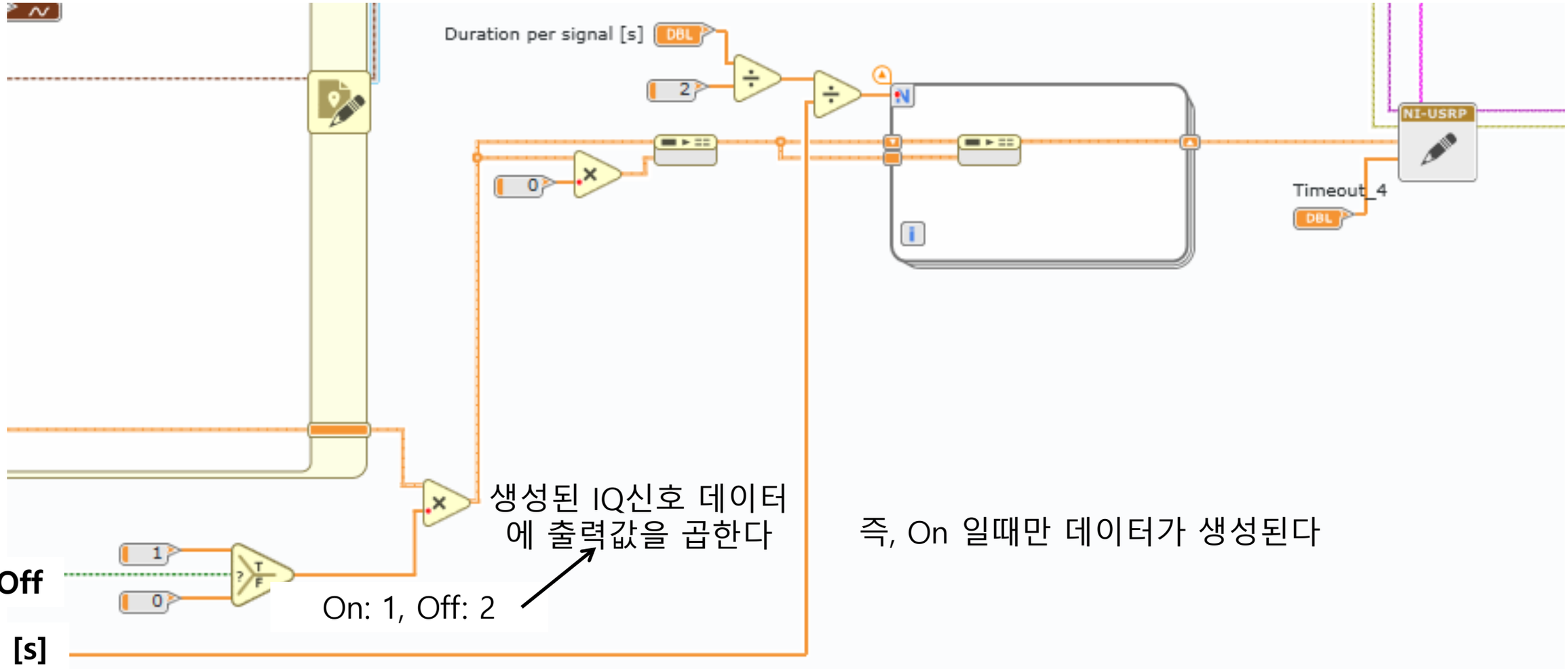
입력한 배열들을 합치는 블록

송신) IQ signal로 바꾸는 부분

- 생성된 Chirp 신호 (시간-주파수 축)를 안테나가 물리적으로 송출할 수 있는 IQ 신호 (시간-Amplitude 축) 로 변환하는 부분

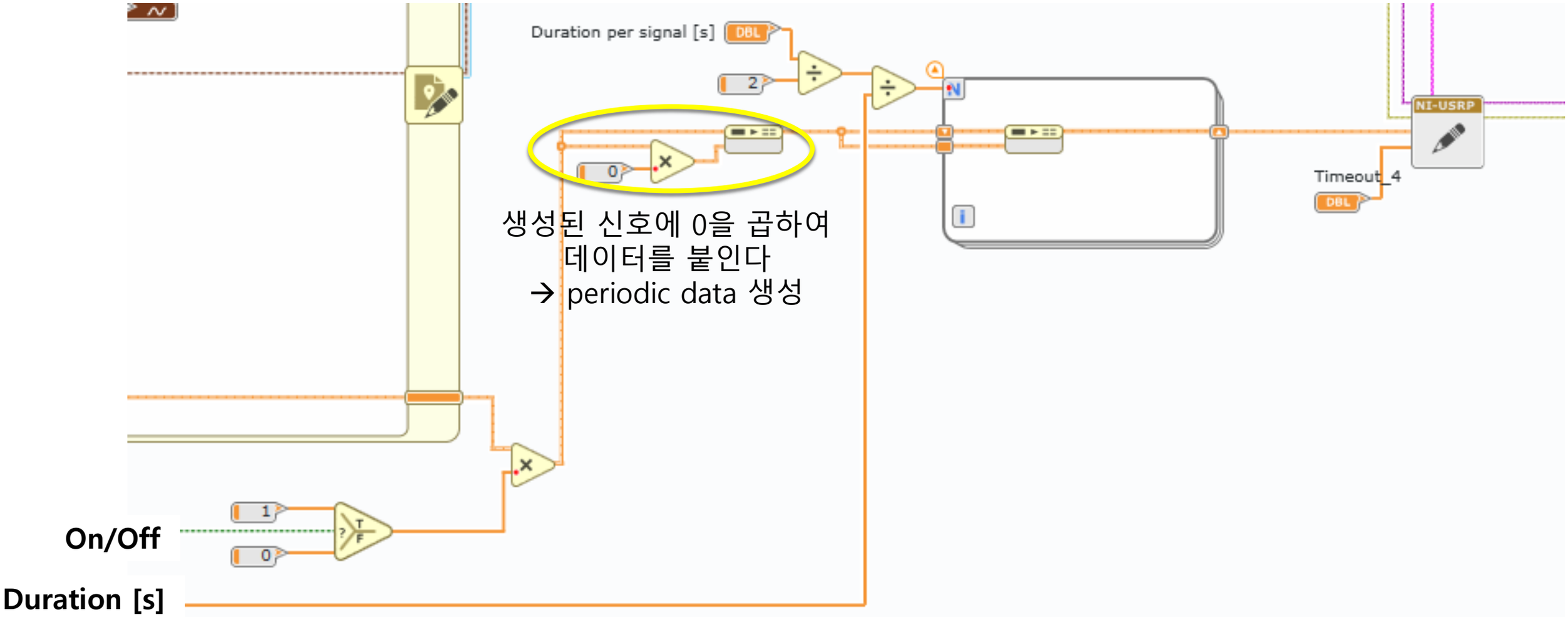


송신) IQ signal을 보내는 부분



즉, On 일때만 데이터가 생성된다

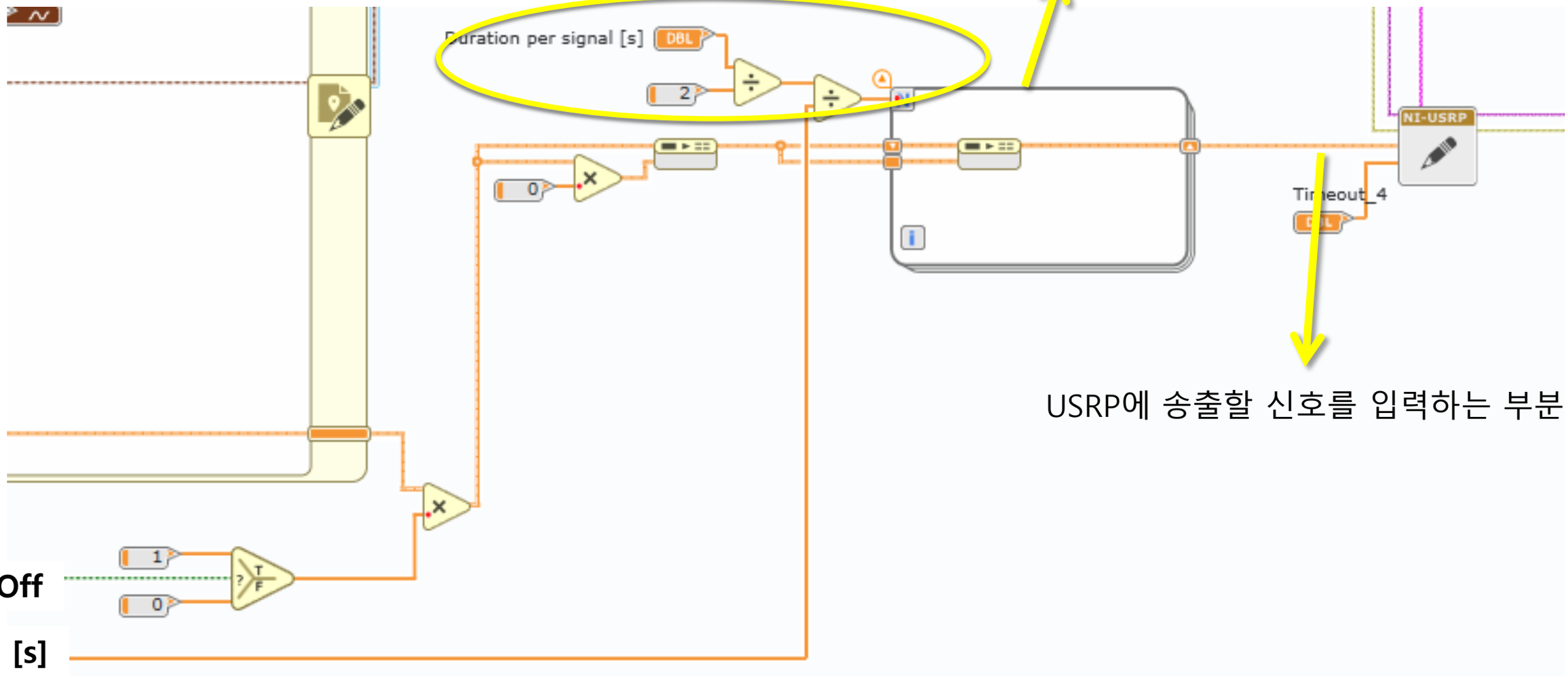
송신) IQ signal을 보내는 부분



송신) IQ signal을 보내는 부분

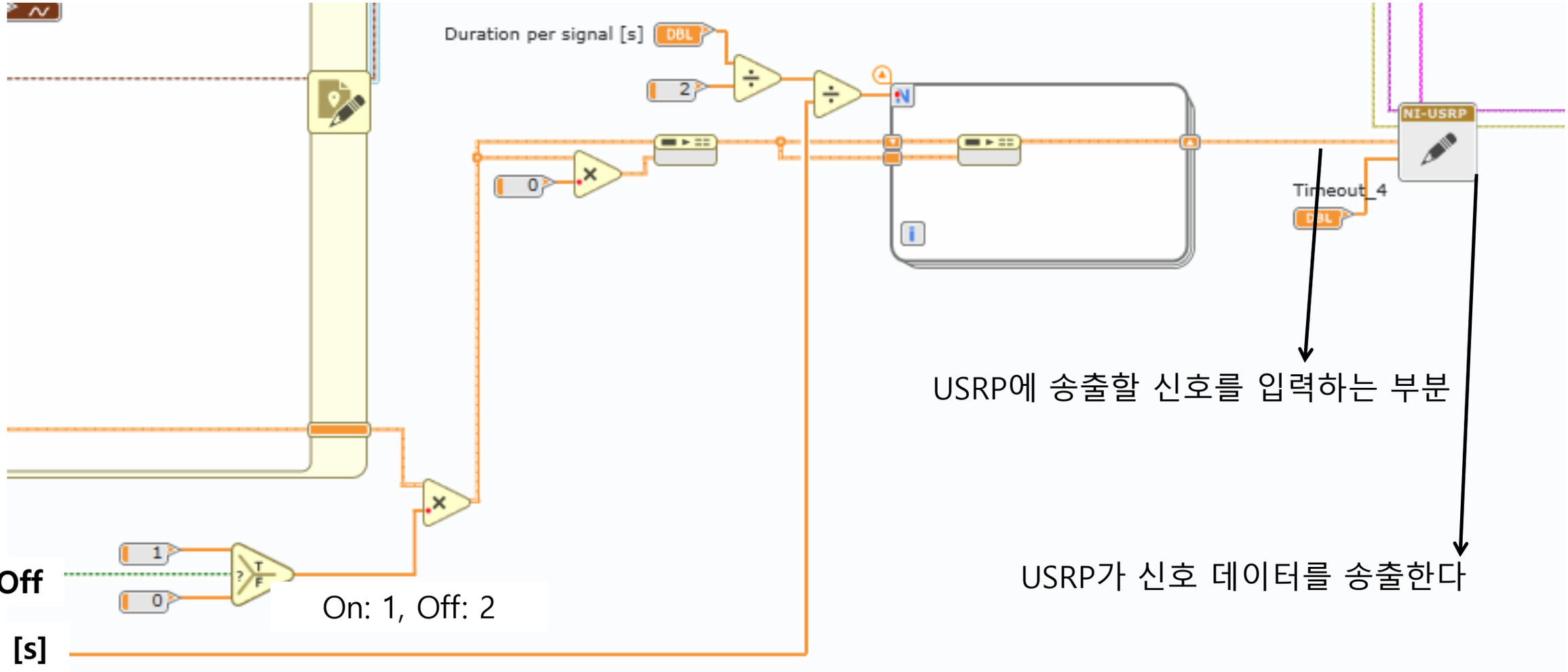
총 실행시간(Duration per signal) 에
period (Duration)이
몇 개 들어가는지를 계산하여

그만큼 For문을 실행하여
생성한 (Duration 만큼의) 신호를
반복하여 붙인다



USRP에 송출할 신호를 입력하는 부분

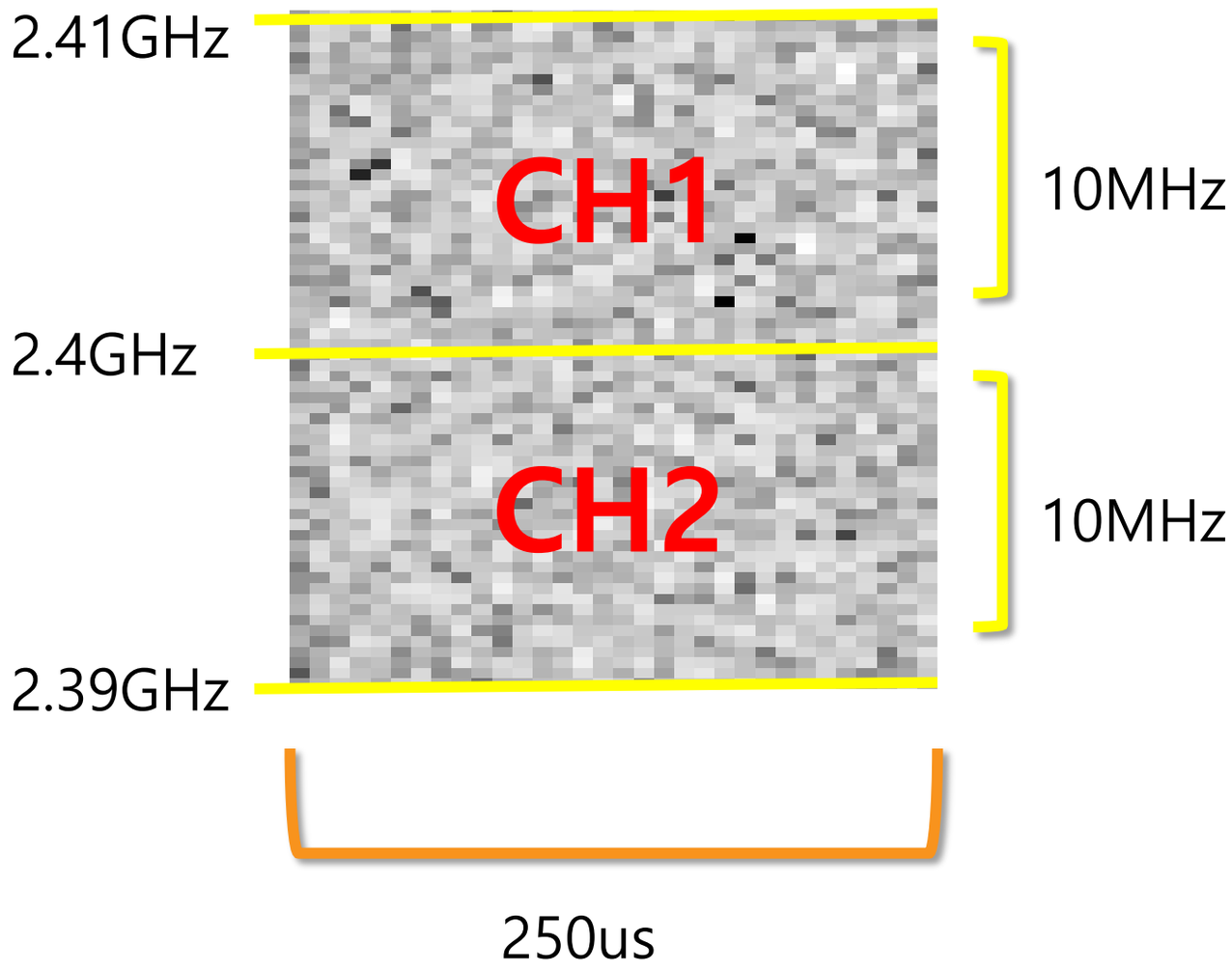
송신) IQ signal을 보내는 부분



송신) 입력값에 관하여

❖ 수신 신호의 visualization (Spectrogram)

- **Spectrogram 설명**

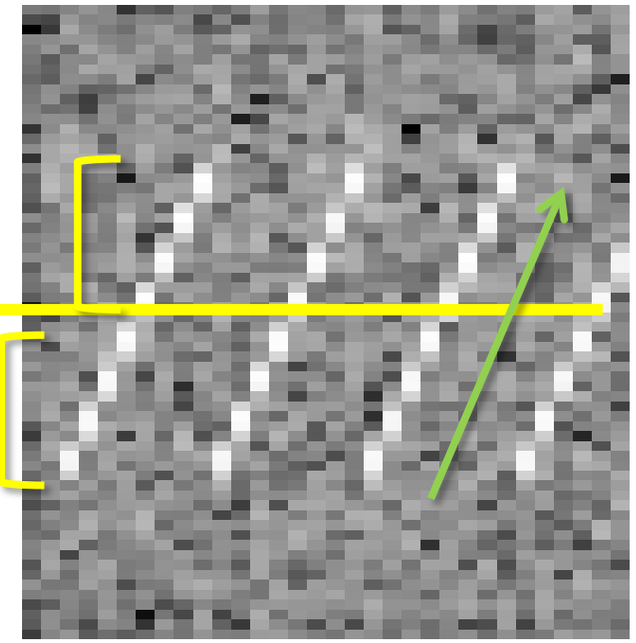


송신) 입력값에 관하여

❖ 수신 신호의 visualization (Spectrogram)

- **본 그림은 왼쪽과 같이 입력했을 때의 결과 스펙트로그램 중 하나의 예시입니다**

Duration per signal	30
Duration	250u
Num of chirps	4
On/off	On
Chirp Direction	Off(Up)
Carrier Frequency	2.4G
IQ rate	10M



Chirp가 2.4GHz를 기준으로 5MHz 씩 위아래 높이로 그려져 있음을 볼 수 있다

Chirp가 위로 (**Off(Up)**) 향하는 대각선임을 볼 수 있다

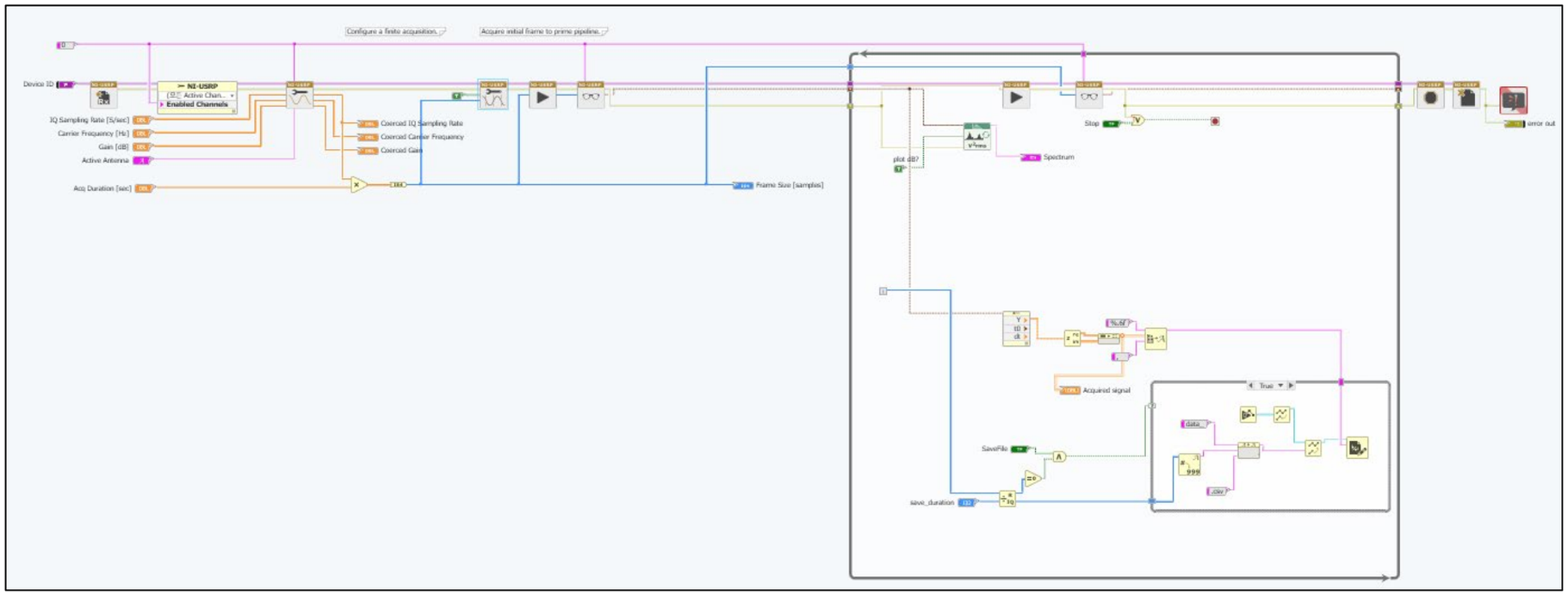
250us(**Duration per signal**) 동안 4개가 (**Num of chirps**) 그려져 있음을 볼 수 있다

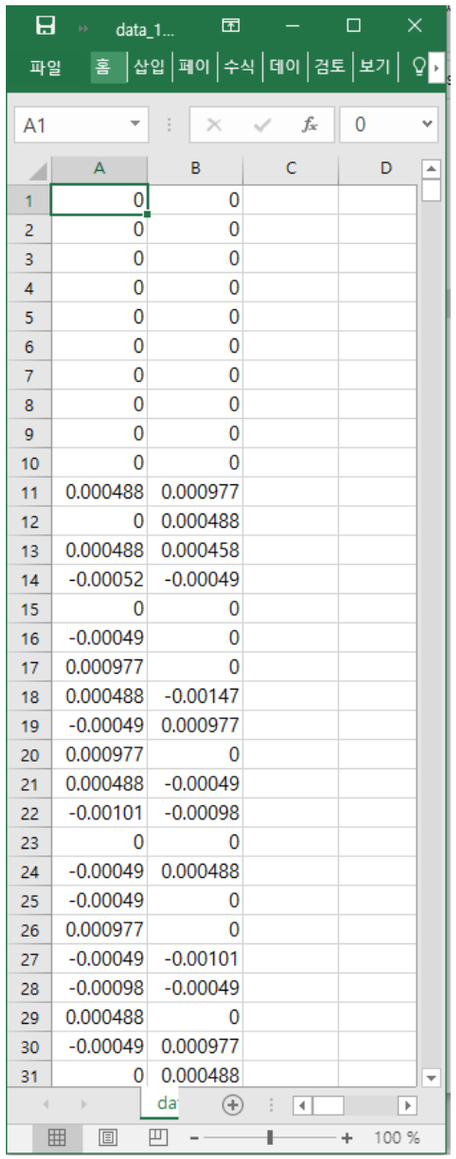
송신) 입력값에 관하여 ※주의

- Duration
  - **250u 보다 작게 하기: 250u로 하면 'Duration per signal'동안 생성되는 데이터가 모두 똑같게 됩니다.**
  - **너무 작게 (10-30u) 설정하면 프로그램 오류가 날 수 있음**
- On/off
  - **항상 On 으로 (파란색)**
- Num of chirps
  - **너무 크게 (15-) 설정하면 프로그램 오류가 날 수 있음**

수신 파트

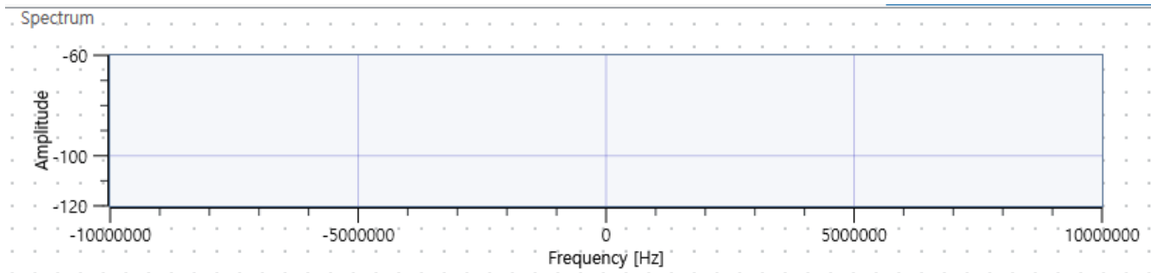
- USRP가 수신하는 IQ신호를 data_(번호).csv 파일로 실시간 저장한다
- 받은신호를 250us마다 하나의 csv 파일로 저장한다
- csv 파일은 Imaginary, Real 값의 두 열로 이루어져 있다





수신) 패널 기본값

- 모든 인풋값을 아래와 똑같이 설정해야 함
  - **Acq Duration:**
    - 스펙트로그램의 x축 (시간)
    - 250u
  - **Carrier Frequency:**
    - 수신 채널의 중심 주파수
    - 2.4G
  - **SaveFile**
    - On: 수신한 IQ signal 데이터를 csv 파일 형식으로 상위 폴더에 저장한다



주파수별 수신한 신호파형

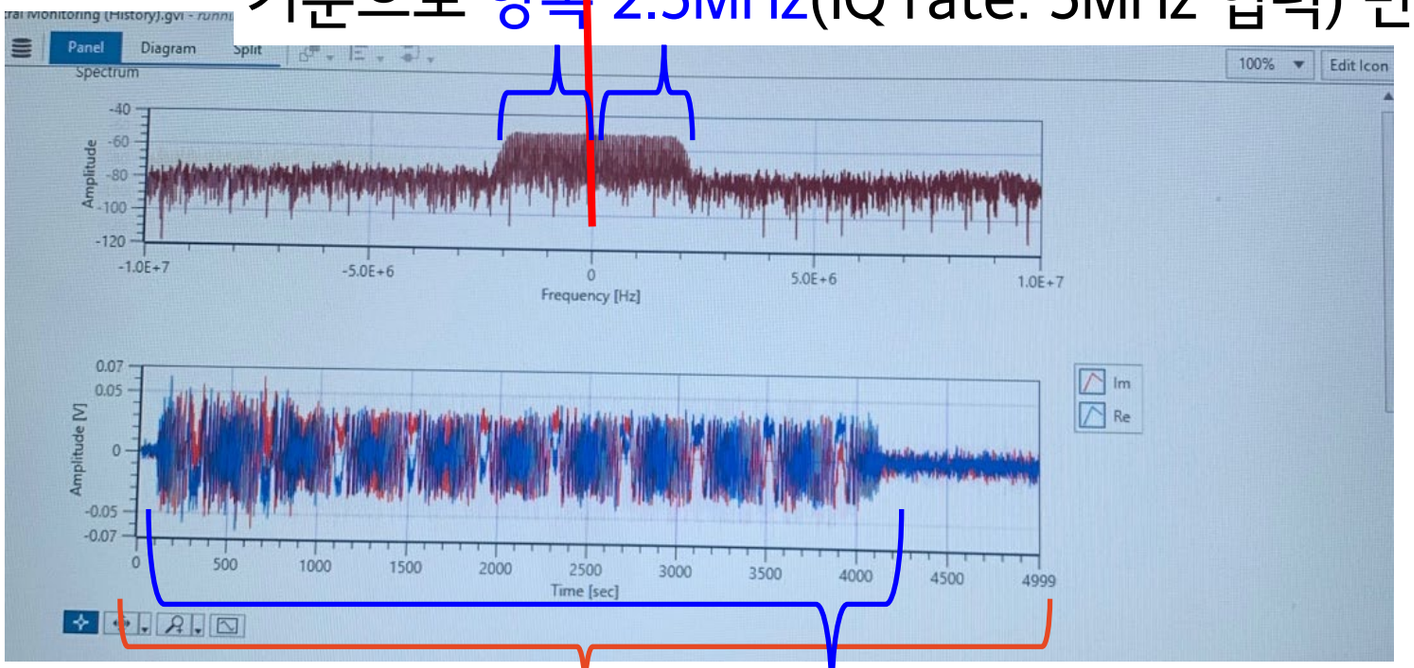
시간별 수신한 신호파형

실행 방법

- 각 조에 두개의 컴퓨터가 할당되어 있음.
- 하나의 컴퓨터에서 TX, 다른 컴퓨터에서 RX 실행
  - **즉 각각 컴퓨터에 각 코드폴더를 저장한다**
  - **특정 컴퓨터에 TX 를 실행해야한다 등의 제약은 없음**

- 신호 송수신이 잘 일어나는지 확인하는 방법 (RX코드의 패널 창)

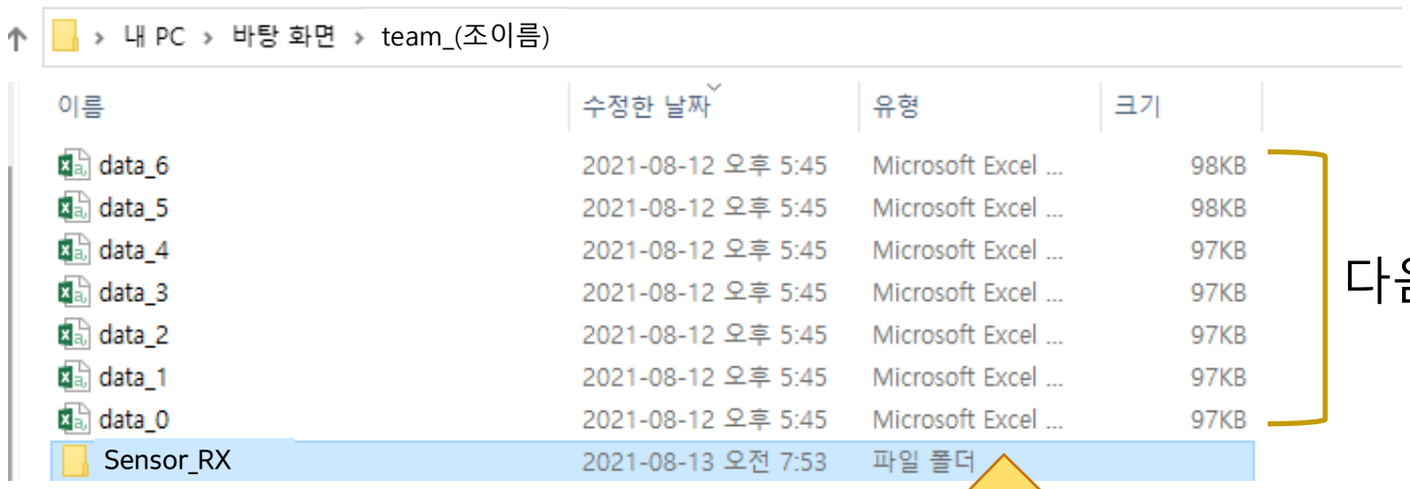
바의 가운데는 2.4GHz를 나타내는데, 입력으로 준 2.4G(Carrier Frequency)
기준으로 양쪽 2.5MHz(IQ rate: 5MHz 입력) 만큼 신호가 나타난다



250us 중 80%정도만(200us(Duration입력)) 동안만 신호가 나타난다

수신) csv파일 저장 위치

- 수신RX) 실행하는 labview 코드가 있는 파일의 상위폴더에 저장된다
  - **즉, team_(조이름) 폴더에 저장됨**



다음과 같이 저장된다

주의사항!!!

- **생성된 data csv 파일 혹은 수정한 코드는 꼭 개인 USB혹은 개인메일로 백업하세요**
  - **컴퓨터를 끄면 모든 데이터가 삭제됩니다**

- 코드를 수정하여 저장하고 싶다면 전체 폴더를 백업하세요
  - **Radar_TX , Sensor_RX 이 폴더 자체를 백업해야 합니다**

과제 (결과 report)

❖ **결과리포트 1**: 예시를 따라 실행. 예시와 같은 인풋값을 입력하여 같은 그래프가 나오는지 확인

- **IQ데이터(csv) 파일과 spectrogram 압축하여 제출**

❖ **결과리포트 2**: 데이터셋 만들기

- **인풋 배열을 10개 이상 지정, Duration per signal 은 40초로 하여 TX를 실행, 동시에 RX를 실행하며 생성된 csv 파일을 잘 저장해놓기**
- **각 채널 1(2.39-2.4GHz) 안/ 채널2 (2.4-2.41GHz) 안/ 채널1과 2에 겹치게 : 각각 생성하여**

  각각 CH1, CH2, both ch 이라는 폴더에 저장 (visualization 후 spectrogram 도 이렇게 폴더 생성)
- **가능하면 아래의 Bonus 문제와 같이 실행하시면 좋습니다**
- **코드 제출, IQ데이터(csv) 와 spectrogram 데이터 압축 제출**

❖ Bonus. 다양한 인풋값에 대하여 실행한 결과를 분석 (워드 or 한글 or ppt 보고서)

- 예시보다 한 스펙트로그램에 더 많은 chirp가 있는 결과 파형
- 예시보다 더 chirp기울기가 원만한 결과 파형
- 예시보다 Bandwidth가 더 넓은 결과 파형

- **각 세부문항에 대해 얻은 spectrogram 그림을 2개씩만 뽑아 보고서에 붙여넣고, 각각 인풋값 작성, 인풋값에 따라 결과가 잘 나온 이유에 대해 간단히 분석, 설명하여 보고서 제출**

예비 보고서

❖ 5주차 예비보고서

1. Supervised Learning 의 training 원리에 대해 간단히 조사하고 설명하시오.
2. CNN 의 구조와 장점에 대해 서술하시오.
3. 데이터셋을 Train / Test / Valid 로 나누는 이유를 설명하시오.
4. 1차 사용자의 스펙트럼 사용 유무를 감지하는 machine learning model 의 “labeling ” 방법을 제안하시오.
5. 문의사항은 이현우 조교에게 연락. (010-4099-9894)