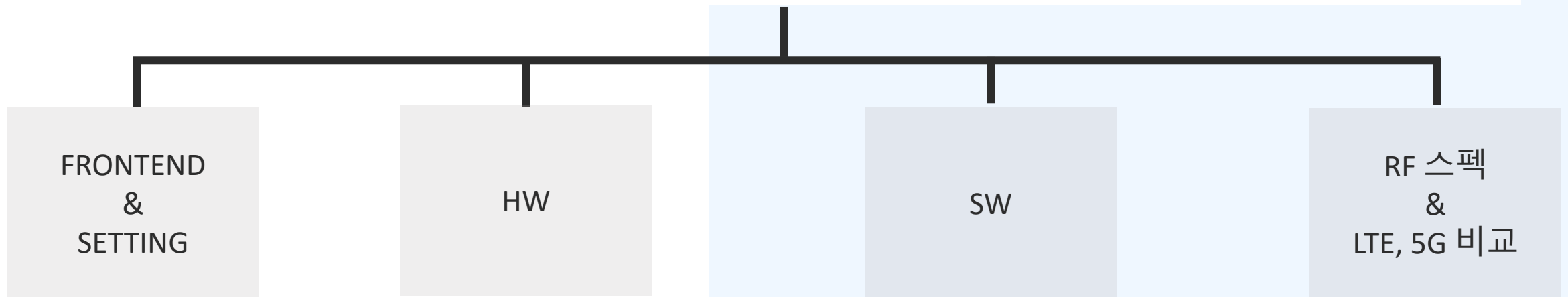
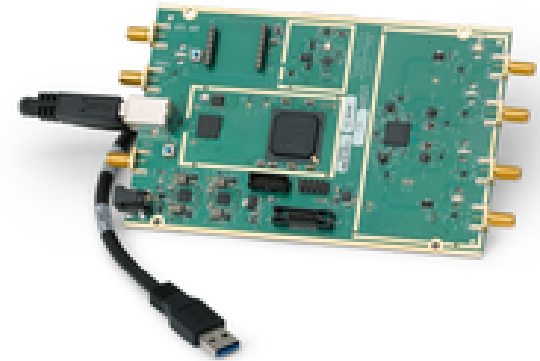


USRP B210  
SPEC  
&  
주파수, 대역폭

김상은

## B210

- Xilinx Spartan 6 XC6SLX150 FPGA
- Analog Devices AD9361 RFIC direct-conversion transceiver
- Frequency range: 70 MHz – 6 GHz
- Up to 56 MHz of instantaneous bandwidth (61.44MS/s quadrature)
- Full duplex, MIMO (2 Tx & 2 Rx)
- Fast and convenient bus-powered USB 3.0 connectivity
- Optional Board Mounted GPSDO



## Operating Environment

Ambient temperature range	23 °C ± 5 °C
Relative humidity range	10% to 90%, noncondensing (tested in accordance with IEC 60068-2-56.)

## Power

Total power, typical operation	
Typical	3 W to 3.5 W
Maximum	4.5 W
Power requirement	Accepts a 6 V, 3 A external DC power connector



- Input / Output Impedance
- All RF Ports are matched to 50 Ohm with -10dB or better return loss generally. Detailed test is pending.

## Interfaces and Connectivity

B200/B210/B200mini - USB 3.0

= > USB 2.0은 Data rate이 느리기 때문에, 적어도 8MS/S 되는 USB 3.0을 사용함.

## § TX/RX 와 RX2 LED 의미

	= 전원이 꺼진 상태
	= RX 상태
(양방향 통신) 	= TX/RX switching 상태
	= TX 상태

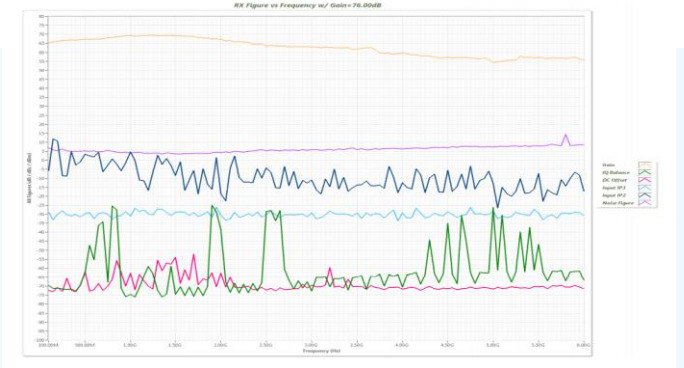
## 1) Tuning

: On the B210, both transmit and receive can be used in a MIMO configuration. For the MIMO case, both receive frontends share the RX LO, and both transmit frontends share the TX LO. Each LO is tunable between 50 MHz and 6 GHz.

## 2) Gains

: All frontends have individual analog gain controls. The receive frontends have 76 dB of available gain; and the transmit frontends have 89.8 dB of available gain. Gain settings are application specific, but it is recommended that users consider using at least half of the available gain to get reasonable dynamic range.

+ RF Performance 참고 : [Test Report \(ettus.com\)](http://ettus.com)



## 3) Bandwidths

: The analog frontend has a seamlessly adjustable bandwidth of 200 kHz to 56 MHz.

Generally, when requesting any possible master clock rate, UHD will automatically configure the analog filters to avoid any aliasing (RX) or out-of-band emissions whilst letting through the cleanest possible signal.

If you, however, happen to have a very strong interferer within half the master clock rate of your RX LO frequency, you might want to reduce this analog bandwidth. You can do so by calling `uhd::usrp::multi_usrp::set_rx_bandwidth(bw)`.

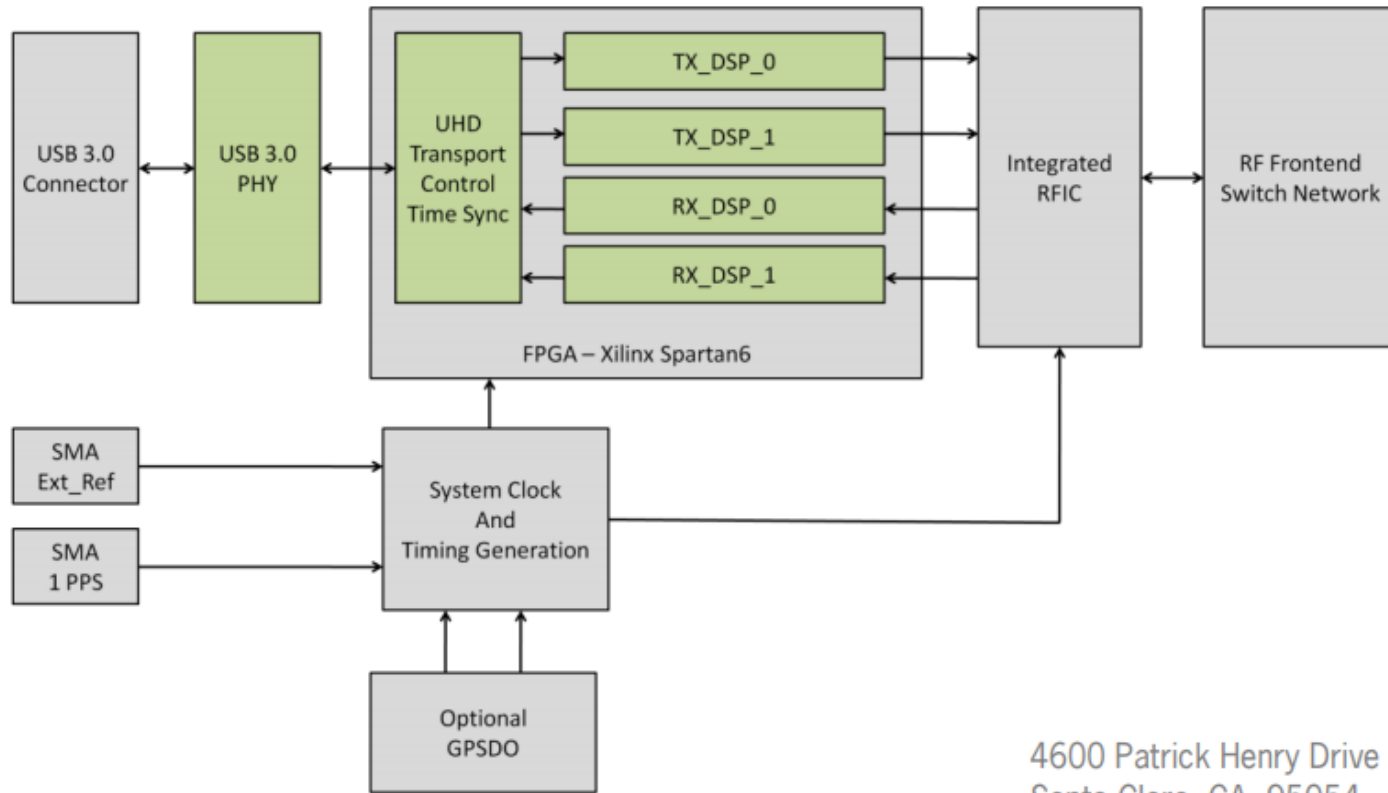
The property to control the analog RX bandwidth is `bandwidth/value`.

UHD will not allow you to set bandwidths larger than your current master clock rate.

## B210

Current Hardware Revision: 5

Minimum version of UHD(USRP HW Drive) required: 3.6.0



4600 Patrick Henry Drive  
Santa Clara, CA 95054

## B210

Device utilization summary:

Selected Device : 6slx150fgg484-3

Slice Logic Utilization:

Number of Slice Registers:	29310	out of	184304	15%
Number of Slice LUTs:	36486	out of	92152	39%
Number used as Logic:	29279	out of	92152	31%
Number used as Memory:	7207	out of	21680	33%
Number used as RAM:	1752			
Number used as SRL:	5455			

Slice Logic Distribution:

Number of LUT Flip Flop pairs used:	43635			
Number with an unused Flip Flop:	14325	out of	43635	32%
Number with an unused LUT:	7149	out of	43635	16%
Number of fully used LUT-FF pairs:	22161	out of	43635	50%
Number of unique control sets:	723			

IO Utilization:

Number of IOs:	180			
Number of bonded IOBs:	163	out of	338	48%
IOB Flip Flops/Latches:	148			

Specific Feature Utilization:

Number of Block RAM/FIFO:	186	out of	268	69%
Number using Block RAM only:	186			
Number of BUFG/BUFGCTRLs:	4	out of	16	25%
Number of DSP48A1s:	152	out of	180	84%

- GNU Radio, C++ and Python APIs 기능 제공
- MATLAB Communication tool을 이용해 USRP분석가능

## GNU(*GNU's Not Unix!*)

⇒ 유닉스와 비슷한 운영체제로서, 복사, 수정 및 재배포가 가능하도록 소스코드가 함께 수록된 것을 말한다. 이것을 이용, 리눅스를 만들었다. gcc(C 컴파일러), gdb(디버거), GNU 체스 등 을 제공함.

## GNU 라디오

위키백과, 우리 모두의 백과사전.

**GNU 라디오**(GNU Radio)는 소프트웨어에 기반한 무선 통신 시스템을 연구하고, 만들고, 제작하기 위한 툴킷이다. 1998년에 시작되었고 지금은 GNU의 정규 프로젝트이다. 존 길모어가 \$32,000의 자금을 조성해서 에릭 블로섬에게 코드 작성과 프로젝트 관리 책임을 일임했다.

GNU Radio는 GPL 하에 배포되는 신호 처리 패키지이다. 목표는 일반적인 소프트웨어 종사자들에게 전자기 스펙트럼을 '해킹'할 수 있도록 하여 무선 스펙트럼을 이해하고 그것을 쉽사리 사용할 수 있게 해 주는 것이다.

다른 모든 소프트웨어 기반 무선 시스템과 같이 재구성 가능성이 핵심 기능이다. 여러개의 값비싼 라디오를 구입하는 것이 아니라 하나의 범용 무선장치만을 구입하여 강력한 신호처리 소프트웨어 (이 경우에는 GNU Radio 이다)로 처리하는 발상이다. 현재는 적은 수의 무선 신호의 형태만을 지원하지만 무선 송신 규칙만 이해한다면 GNU Radio를 이용해 수신 가능하도록 구성할 수 있다.

GNU Radio는 MIT의 Pspectra 프로젝트에서 개발된 코드로부터 파생되었다. Pspectra SDR 설계는 파이썬 언어의 강력한 모듈식 파이프라인과 수월한 프로그래밍 기법을 이용하도록 만들어졌다. 2004년에 GNU Radio는 완전히 "재작성"되었지만 상당한 부분의 원래 Pspectra 코드와 구조는 그대로 남아 있다. 상용 Vanu 소프트웨어 라디오가 Pspectra 의 코드에 기반하고 있다.

GNU Radio 프로젝트에서 USRP(Universal Software Radio Peripheral)이 개발되었다. 이것은 저렴한 4채널 20 MSP DAQ이다. 몇 개의 라디오 모듈을 이용할 수 있다. USRP는 매트 에투스가 개발하였다.

### GNU 라디오 GNU Radio



원저자	에릭 블로섬
개발자	GNU 프로젝트 Tom Rondeau Johnathan Corgan
안정화 버전	3.9.1.0 <sup>[1]</sup> / 2021년 3월 22일 (9일 전)
저장소	<a href="https://github.com/gnuradio/gnuradio.git">github.com/gnuradio/gnuradio.git</a>
프로그래밍 언어	C++, 파이썬
운영 체제	크로스 플랫폼
언어	영어
종류	라디오
라이선스	GNU GPL

## Transmitter

Frequency range	70 MHz to 6 GHz
Frequency step	<1 kHz
Maximum output power ( $P_{out}$ )	20 dBm
Gain range <sup>1</sup>	89.75 dB
Gain step	0.25 dB
Frequency accuracy <sup>2</sup>	2.5 ppm
Maximum instantaneous real-time bandwidth	56 MHz
Maximum I/Q rate <sup>3</sup>	
Streaming <sup>4</sup>	15 MS/s
Burst	
One channel	61.44 MS/s
Two channels	30.72 MS/s
Digital-to-analog converter (DAC) <sup>5</sup>	12 bits

## \*Sampling 속도 단위\*

MS/s

- ⇒ Milion sample / sec
- ⇒ 초당 백만 샘플처리
- ⇒ 1us마다 샘플링하는 것

## \*dBm\*

⇒ 전력값을 1mW를 기준으로 dB화 한 값.

1mW = 0dBm

10mW = 10 dBm

100mW = 20dBm

1000mW = 30dBm = 1W

## Receiver

Frequency range	70 MHz to 6 GHz
Frequency step	<1 kHz
Gain range <sup>6</sup>	76 dB
Gain step	1.0 dB
Maximum input power ( $P_{in}$ )	-15 dBm
Noise figure	5 dB to 7 dB
Frequency accuracy <sup>7</sup>	2.5 ppm
Maximum instantaneous real-time bandwidth	56 MHz
Maximum I/Q rate <sup>8</sup>	
Streaming <sup>9</sup>	15 MS/s
Burst	
One channel	61.44 MS/s
Two channels	30.72 MS/s
Analog-to-digital converter (ADC) <sup>10</sup>	12 bits

(단, 모든 spec의 기준은 SISO(1:1)인 상황)

Maximum instantaneous real-time bandwidth	TX /RX
SISO(1× 1)	56 MHz
MIMO(2× 2)	30.72 MHz



## 이동통신 사업자 주파수 할당현황 (2020. 01)

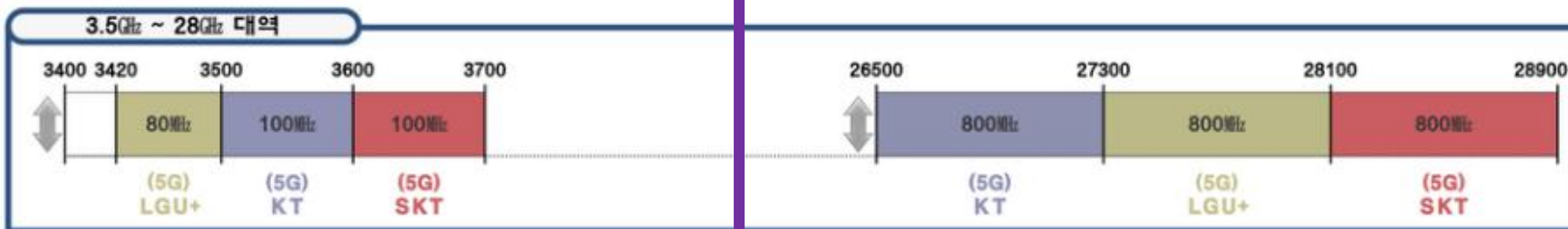
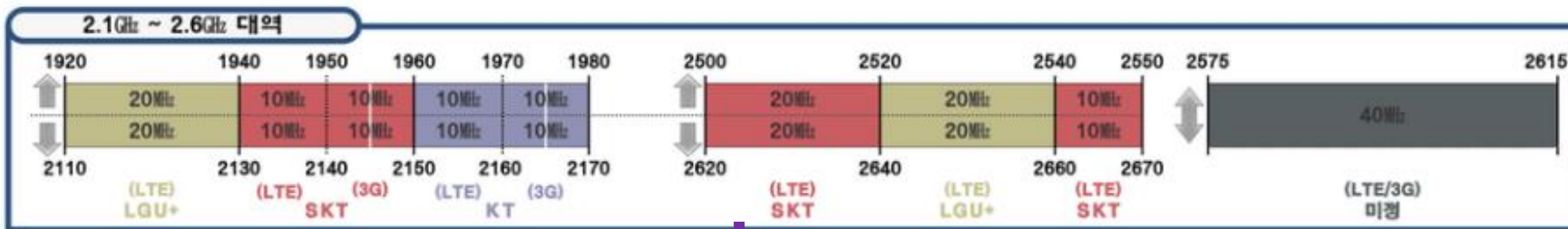
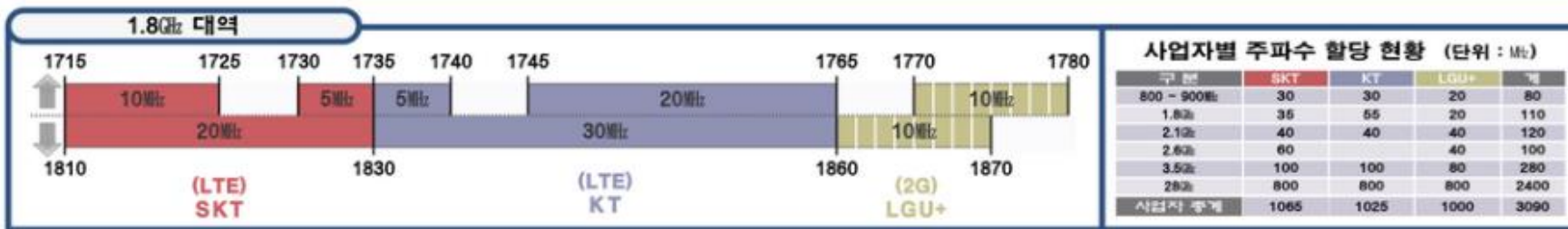
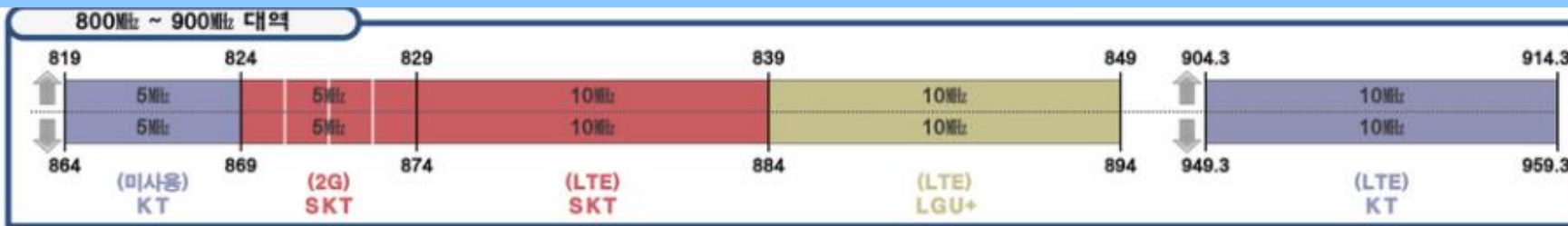
과학기술정보통신부

KCA 한국방송통신전파진흥원

↑ : uplink

↓ : downlink

↕ : TDD



LTE 주파수

(3GPP에서 규정한 LTE대역폭)

1.4 MHz

3 MHz

5 MHz

10 MHz

15 MHz

20 MHz 중 하나여야만 함.

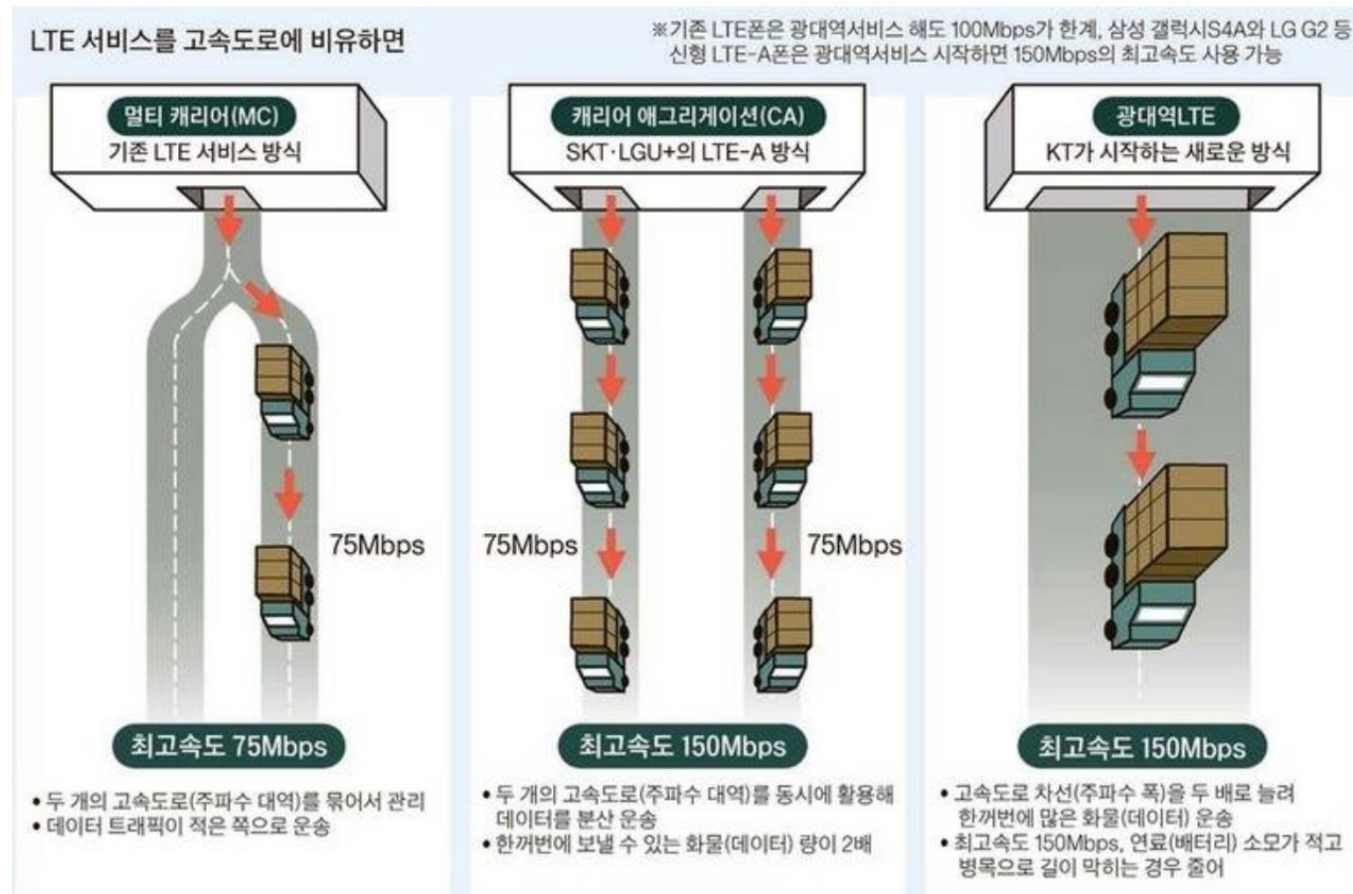
5G(NR) 주파수

최대  
100MHz최대  
400MHz(기준)  
6GHz

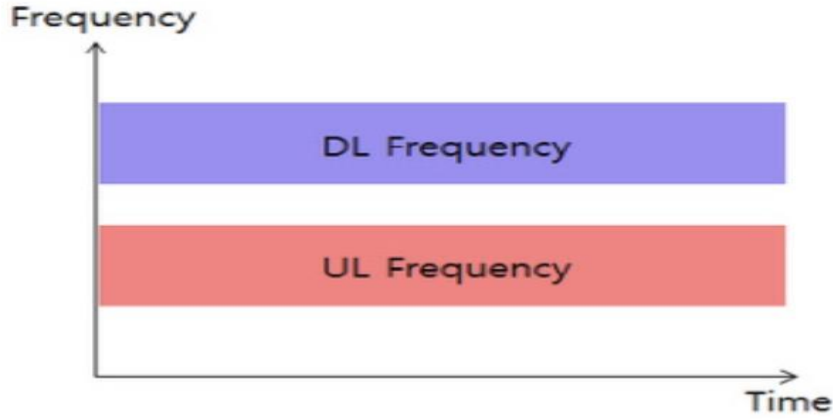
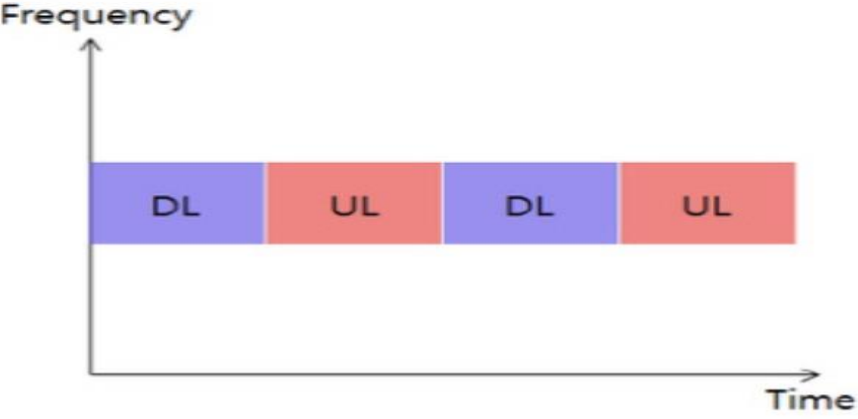


## CA(Carrier Aggregation)

- 서로 다른 대역의 주파수를 하나의 연속적인 대역폭으로 묶어주는 기술
- LTE-A에서 속도를 증가시키기 위한 가장 핵심적인 기술
- 연속적으로 주파수 대역을 확보하는 것이 어려워서 개발됨.



차이점) uplink와 downlink를 어떻게 나눠서 쓰는가

FDD(Frequency Division Duplexing)	TDD(Time Division Duplexing)
<p data-bbox="262 439 1052 486">UL와 DL를 서로 다른 주파수에 배정함.</p> <p data-bbox="606 711 848 753"><b>LTE FDD</b></p> 	<p data-bbox="1274 439 2186 601">UL, DL가 주파수로 구분되어 있지 않고 동일한 주파수 내에서 서로 다른 시간으로만 구분하는 것.</p> <p data-bbox="1600 711 1854 753"><b>LTE TDD</b></p> 

UL, DL는 일정한 이격거리를 두고 쌍으로 할당된다.  
기지국에서 단말로 보내는 하향링크가 더 많은 데이터를 보내기 때문에 통신에서는 하향이 더 많은 구조를 지님.

## nr-ue.c code

```

204: static const eutra_band_t eutra_bands[] = {
205:     { 1, 1920 * MHz, 1980 * MHz, 2110 * MHz, 2170 * MHz, FDD },
206:     { 2, 1850 * MHz, 1910 * MHz, 1930 * MHz, 1990 * MHz, FDD },
207:     { 3, 1710 * MHz, 1785 * MHz, 1805 * MHz, 1880 * MHz, FDD },
208:     { 4, 1710 * MHz, 1755 * MHz, 2110 * MHz, 2155 * MHz, FDD },
209:     { 5, 824 * MHz, 849 * MHz, 869 * MHz, 894 * MHz, FDD },
210:     { 6, 830 * MHz, 840 * MHz, 875 * MHz, 885 * MHz, FDD },
211:     { 7, 2500 * MHz, 2570 * MHz, 2620 * MHz, 2690 * MHz, FDD },
212:     { 8, 880 * MHz, 915 * MHz, 925 * MHz, 960 * MHz, FDD },
213:     { 9, 1749900 * KHz, 1784900 * KHz, 1844900 * KHz, 1879900 * KHz, FDD },
214:     {10, 1710 * MHz, 1770 * MHz, 2110 * MHz, 2170 * MHz, FDD },
215:     {11, 1427900 * KHz, 1452900 * KHz, 1475900 * KHz, 1500900 * KHz, FDD },
216:     {12, 698 * MHz, 716 * MHz, 728 * MHz, 746 * MHz, FDD },
217:     {13, 777 * MHz, 787 * MHz, 746 * MHz, 756 * MHz, FDD },
218:     {14, 788 * MHz, 798 * MHz, 758 * MHz, 768 * MHz, FDD },
219:     {17, 704 * MHz, 716 * MHz, 734 * MHz, 746 * MHz, FDD },
220:     {20, 832 * MHz, 862 * MHz, 791 * MHz, 821 * MHz, FDD },
221:     {22, 3510 * MHz, 3590 * MHz, 3410 * MHz, 3490 * MHz, FDD },
222:     {33, 1900 * MHz, 1920 * MHz, 1900 * MHz, 1920 * MHz, TDD },
223:     {34, 2010 * MHz, 2025 * MHz, 2010 * MHz, 2025 * MHz, TDD },
224:     {35, 1850 * MHz, 1910 * MHz, 1850 * MHz, 1910 * MHz, TDD },
225:     {36, 1930 * MHz, 1990 * MHz, 1930 * MHz, 1990 * MHz, TDD },
226:     {37, 1910 * MHz, 1930 * MHz, 1910 * MHz, 1930 * MHz, TDD },
227:     {38, 2570 * MHz, 2620 * MHz, 2570 * MHz, 2630 * MHz, TDD },
228:     {39, 1880 * MHz, 1920 * MHz, 1880 * MHz, 1920 * MHz, TDD },
229:     {40, 2300 * MHz, 2400 * MHz, 2300 * MHz, 2400 * MHz, TDD },
230:     {41, 2496 * MHz, 2690 * MHz, 2496 * MHz, 2690 * MHz, TDD },
231:     {42, 3400 * MHz, 3600 * MHz, 3400 * MHz, 3600 * MHz, TDD },
232:     {43, 3600 * MHz, 3800 * MHz, 3600 * MHz, 3800 * MHz, TDD },
233:     {44, 703 * MHz, 803 * MHz, 703 * MHz, 803 * MHz, TDD },
234:     {45, 3300 * MHz, 3500 * MHz, 3300 * MHz, 3500 * MHz, TDD },
235:     {99, 2400 * MHz, 2495 * MHz, 2400 * MHz, 2495 * MHz, TDD },
236: };
237:

```

Q1. 코드 구현을 할 때, FDD와 TDD가  
나뉘어 구현이 되어있는데, 어떠한  
기준으로 나뉘진 것인가요?



```

365:
366:
367: #define NHAL_FREQ_3GHZ          3000000000
368: #define NHAL_NARFCN_3GHZ       600000
369:
370: #define NHAL_WATCHDOG_INIT      5000
371: #define NHAL_WATCHDOG_ONOFF     100
372:
373:

```

Q2. 위의 NHAL\_FREQ 가 3GHz이면  
지금 통신할 때, 사용하고 있는  
주파수가 3GHz라는 의미인가요?

Maximum instantaneous real-time bandwidth	56 MHz
---	--------

```

769: #if 1
770:     double rf_sampling_rate = (15360000.0);           // = 7680000
771:     double rf_tx_bw = 10.0e6;                         // 10e6(original)
772:     fftSize = nhal_rxFFT_1024;
773: #else
774:     //7.68MHz sampling (15kHz SCS)
775:     double rf_sampling_rate = (15360000.0/2);         // = 7680000
776:     double rf_tx_bw = 5.0e6;                          // 10e6(original)
777:     fftSize = nhal_rxFFT_512;
778:
779:     //double rf_sampling_rate = samplingRate;
780:     //double rf_tx_bw = rf_sampling_rate / 15.36;
781: #endif
782:
783:     uint32_t samplePerSlot = (int)rf_sampling_rate/1000;
784:     double rf_rx_bw = rf_tx_bw;
785:

```

Q3. Bandwidth는 10MHz를 사용하는 것인가요?  
그렇다면 스펙에서 최대 56MHz라고 되어있는데  
그럼 5G에서 말하는 대역폭인 100MHz 또는 800MHz는  
현재 USRP B210에서는 실험이 불가능한 것인가요?



**THANK  
YOU**