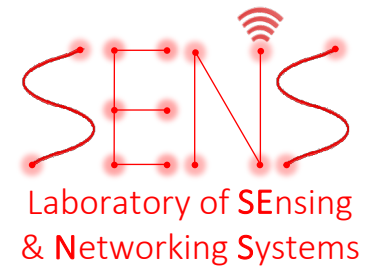
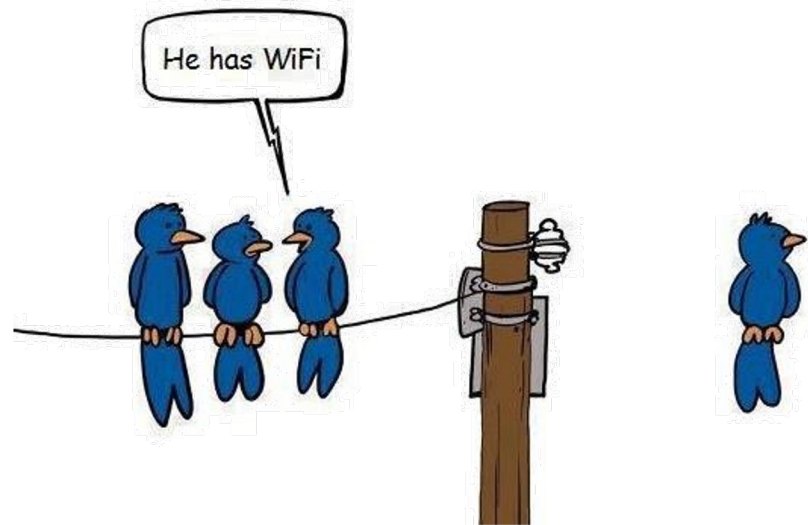


COM-405: Mobile Networks

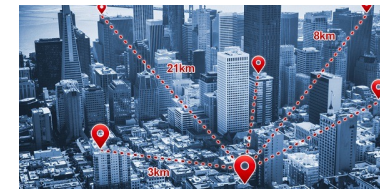
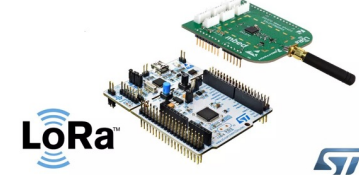
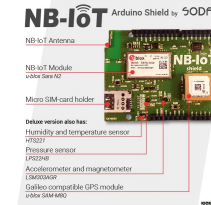
Lecture 9.0: IoT Technologies Haitham Hassanieh



IoT Technologies



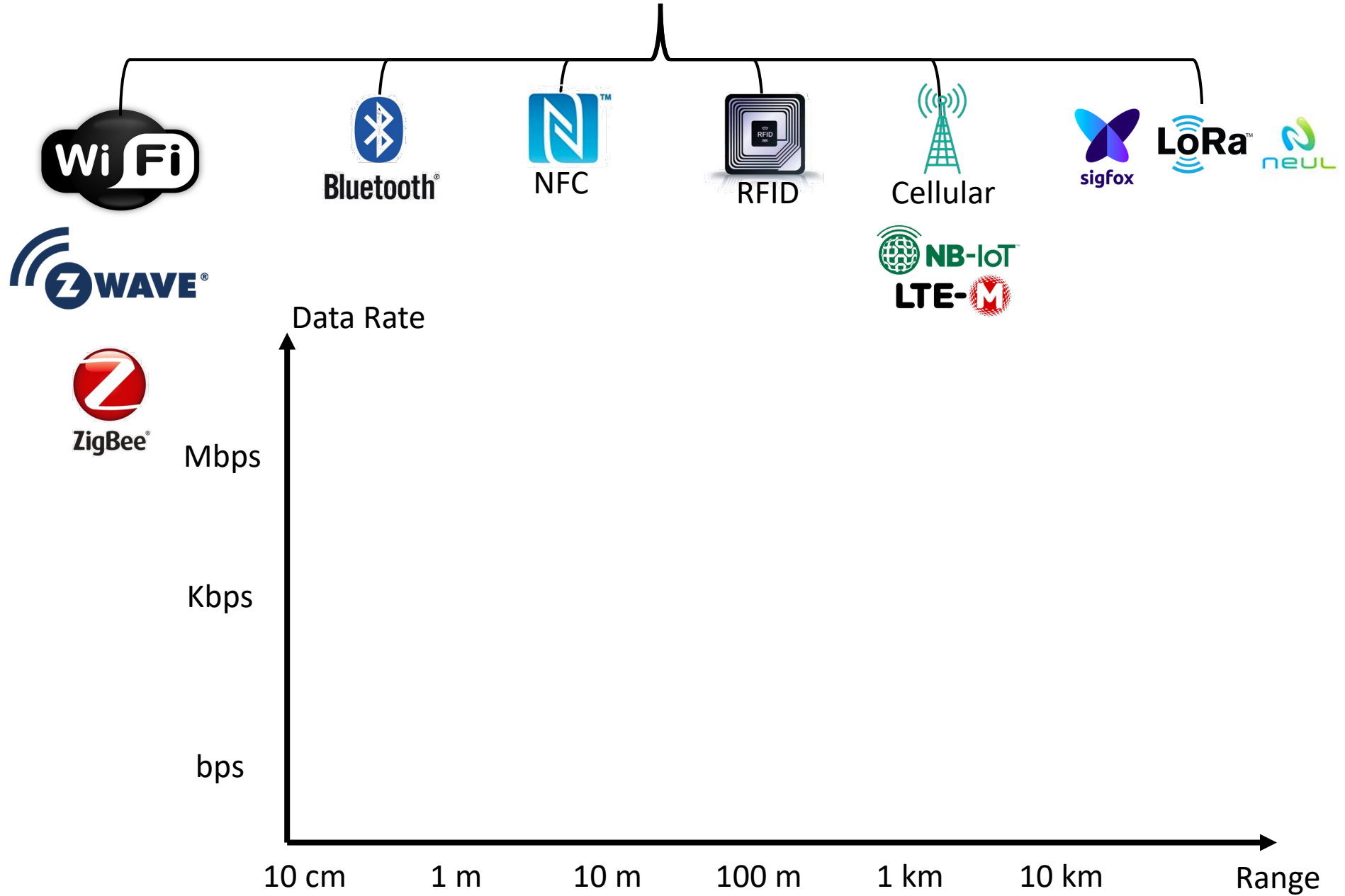
Cellular



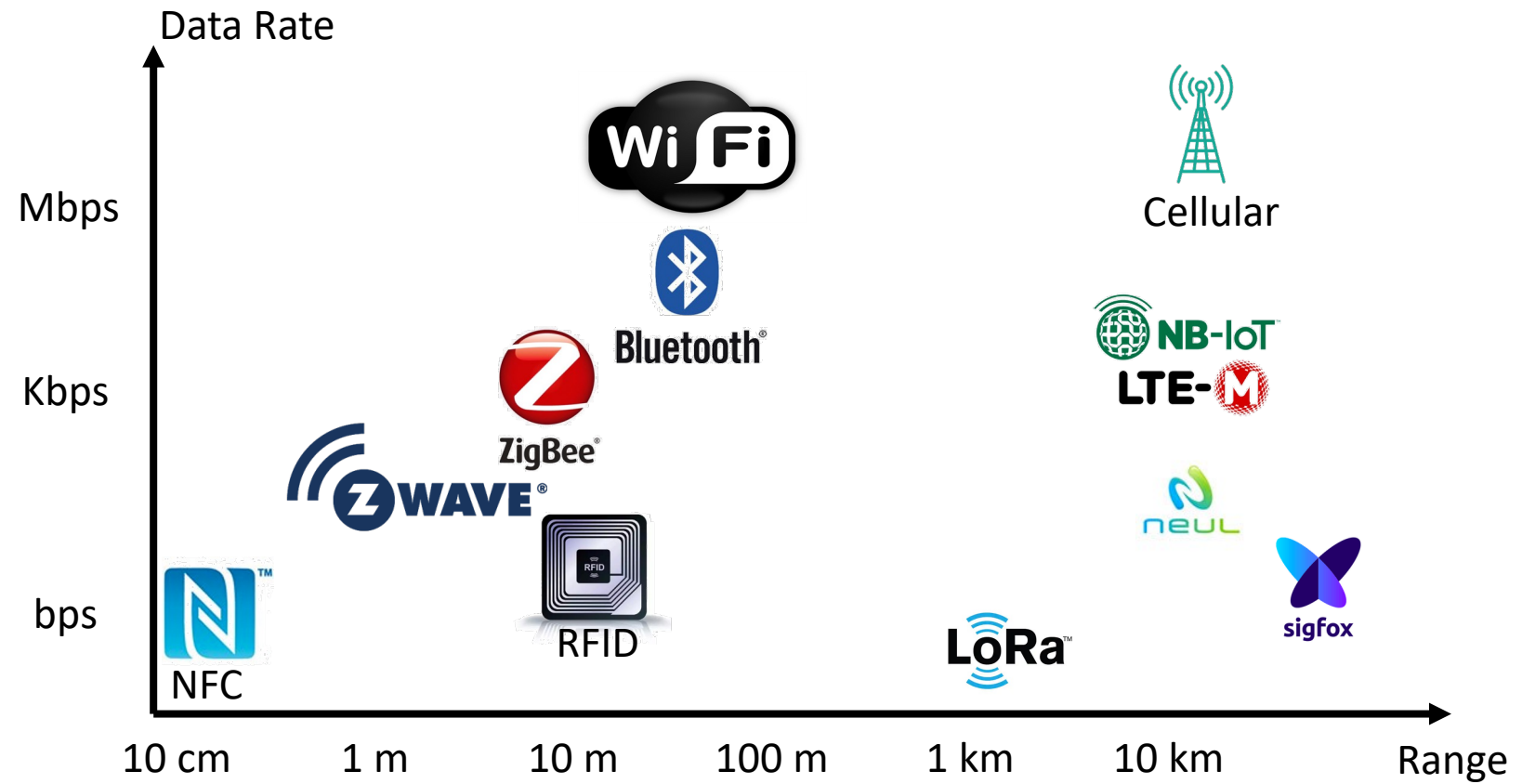
LoRa™ Smart Agriculture Applications



IoT Technologies

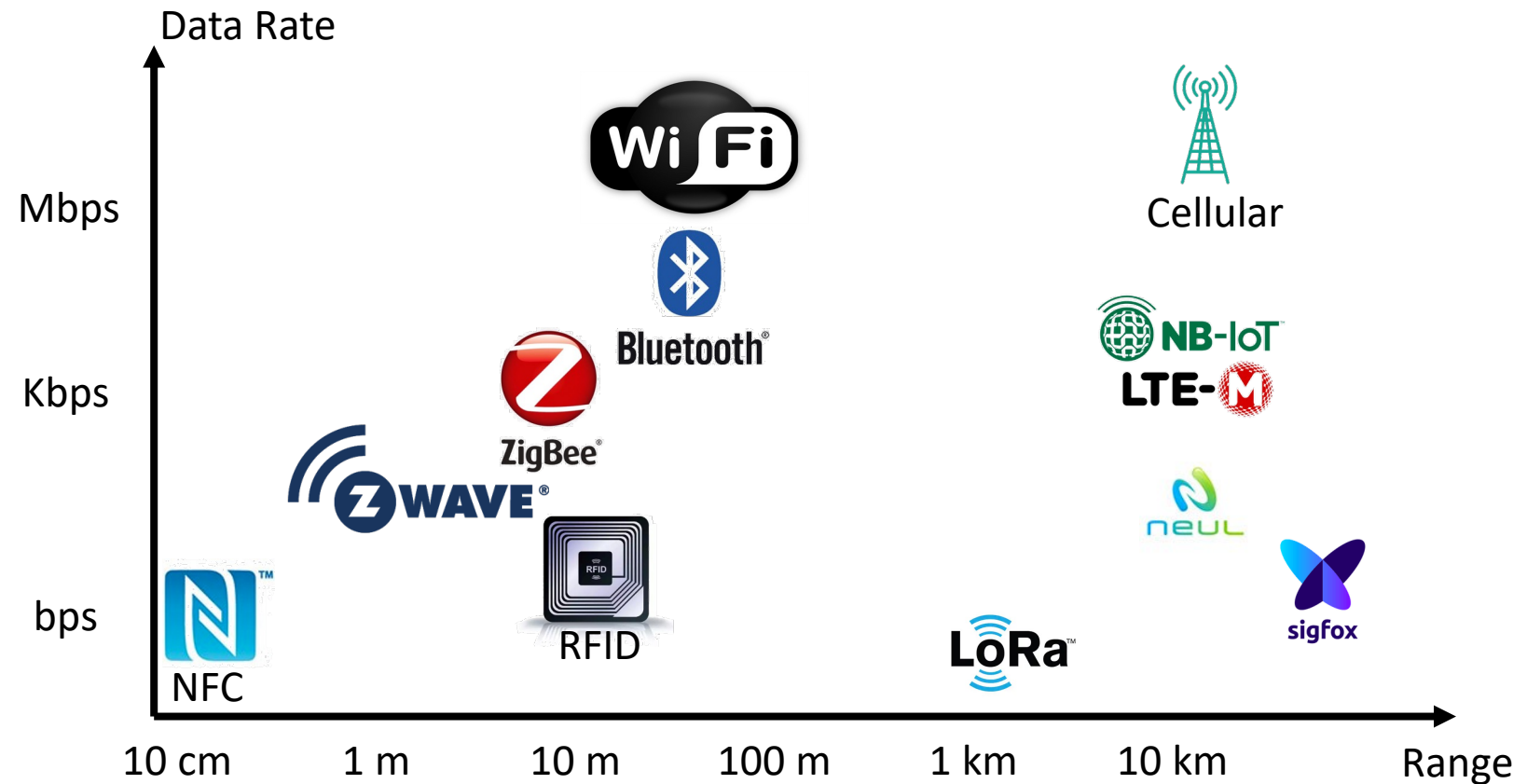


IoT Technologies



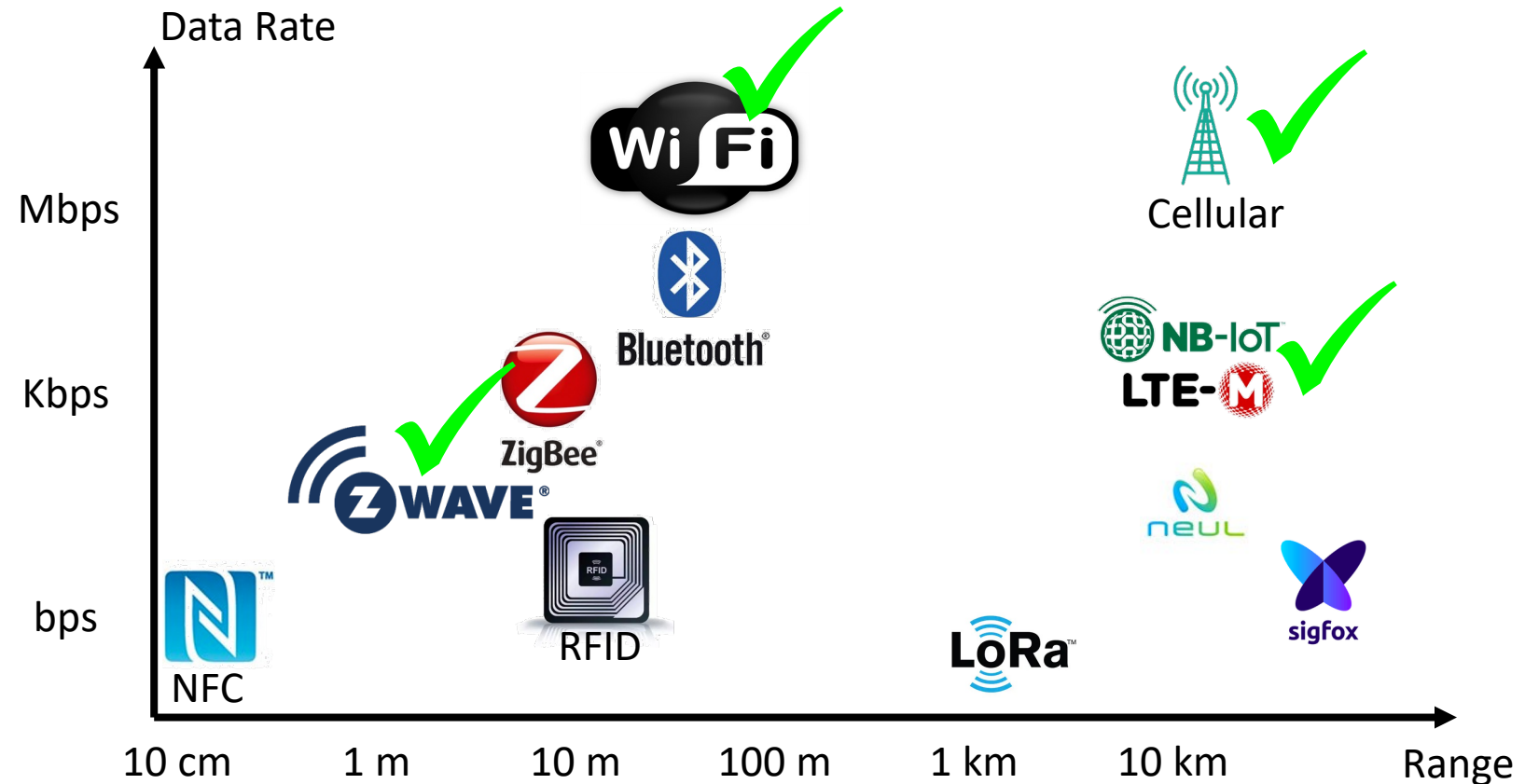
IoT Technologies

Metrics to consider: Data Rate, Range,



IoT Technologies

Metrics to consider: Data Rate, Range, Power (batter life), Cost



IoT: Low Power Wide Area Networks

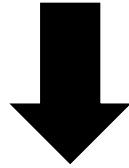


- Low power: battery lasts 5-10 years
- Low cost: \$10-\$20
- High range: 1 - 10 km
- Low Data rates: 100bps – 250 Kbps

IoT: Low Power Wide Area Networks



IoT: Low Power Wide Area Networks



Digital Communication Technology:

CCS: Chirp Spread
Spectrum

CSS: Chirp Spread Spectrum

- Chirp is signal that continuously sweeps frequency with time.
- For linear chirp: $f(t) = \alpha t + f_0$

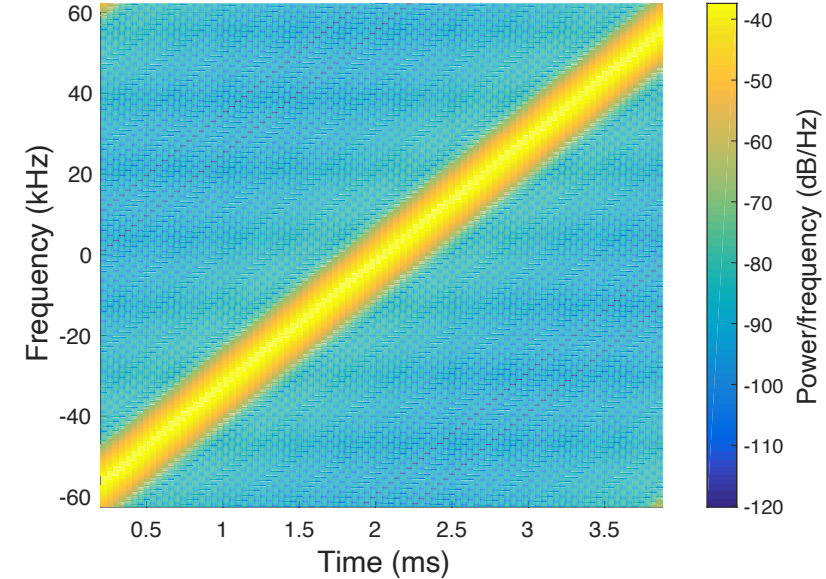
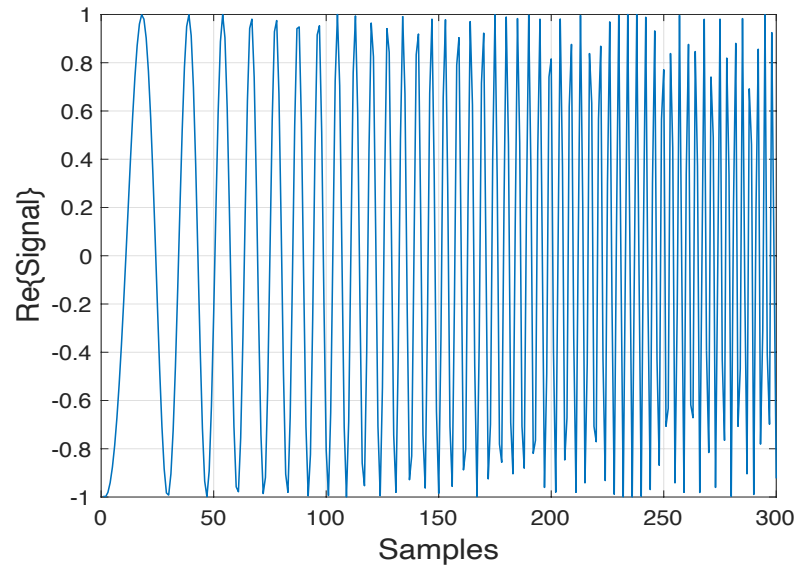
$$\phi(t) = \int 2\pi f(t) dt = 2\pi \frac{\alpha}{2} t^2 + 2\pi f_0 t + \phi_0$$

$$s(t) = \exp\left(j2\pi \frac{\alpha}{2} t^2 + j2\pi f_0 t + j\phi_0\right)$$

CSS: Chirp Spread Spectrum

- For linear chirp: $f(t) = \alpha t + f_0$

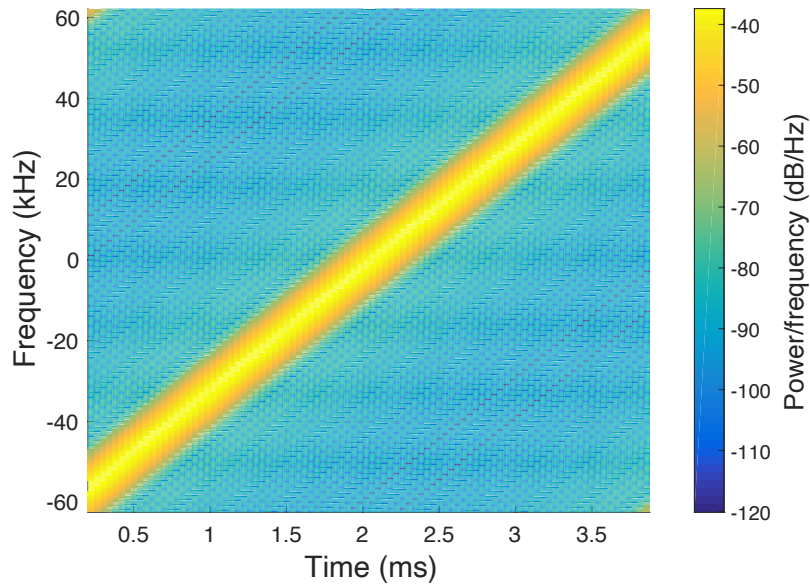
$$s(t) = \exp\left(j2\pi\frac{\alpha}{2}t^2 + j2\pi f_0t + j\phi_0\right)$$



CSS: Chirp Spread Spectrum

- For linear chirp: $f(t) = \alpha t + f_0$

$$s(t) = \exp\left(j2\pi\frac{\alpha}{2}t^2 + j2\pi f_0 t + j\phi_0\right)$$

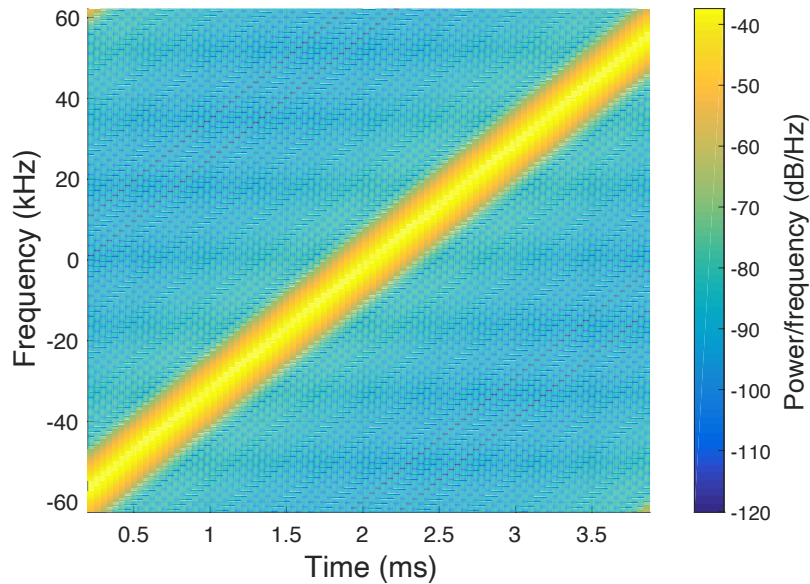


- Sweep from $-\frac{B}{2}$ to $+\frac{B}{2}$
- Sweep time is: T_s
- Sweep slope is: $\alpha = \frac{B}{T_s}$

CSS: Chirp Spread Spectrum

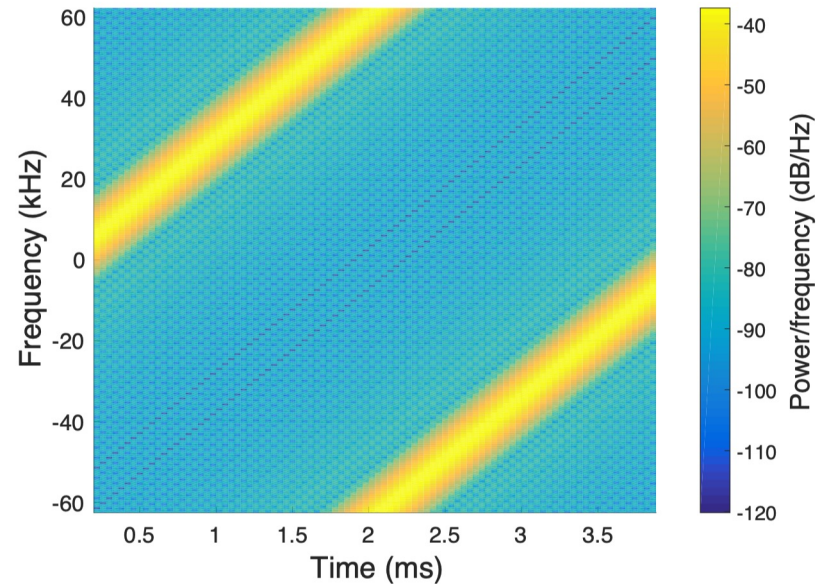
- Each symbol is sent as 1 chirp.
- Symbol time = Sweep time = T_s .
- Encode bits at sweep start.

bit = '0'



$$f(t) = \alpha t - B/2$$

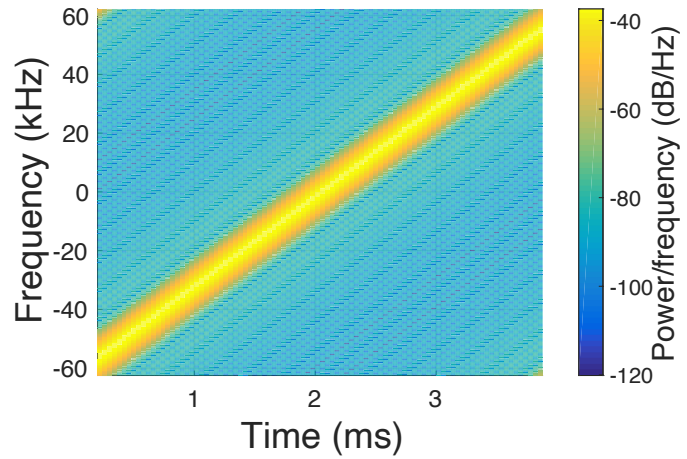
bit = '1'



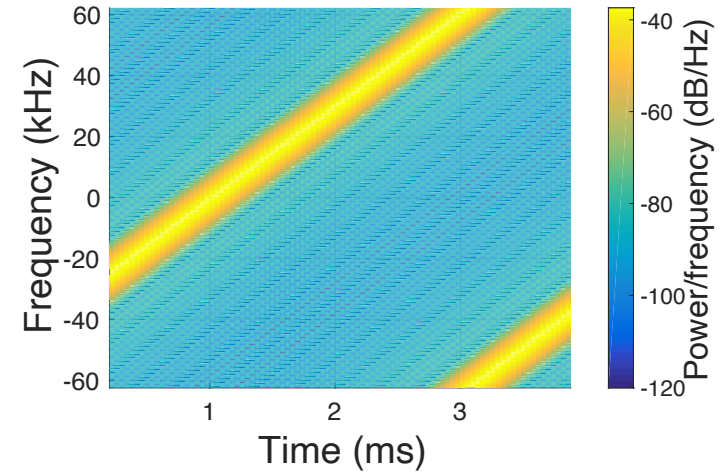
$$f(t) = \alpha t \bmod B$$

CSS: Chirp Spread Spectrum

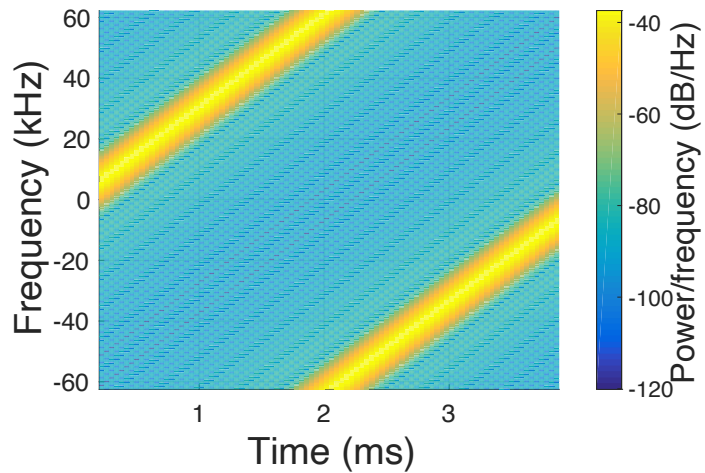
bits = '00'



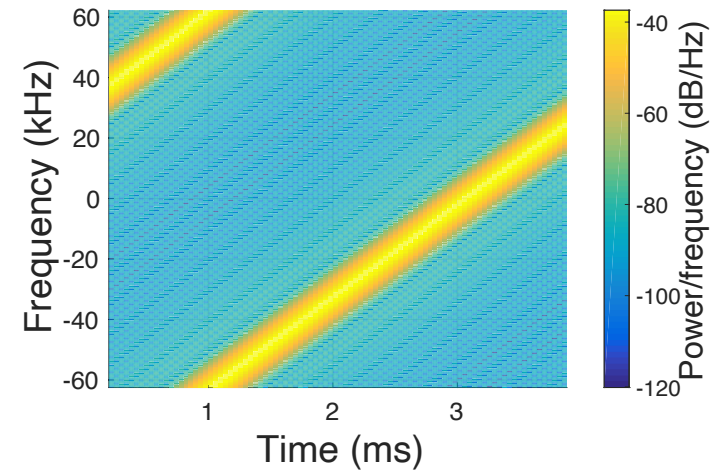
bits = '01'



bits = '10'



bits = '11'



CSS: Chirp Spread Spectrum

In general, to encode n bits per symbol: 2^n shifts

But, How do we decode?

But, How do we decode?

At TX:

bit = '0' $s_0(t) = \exp\left(j2\pi\frac{\alpha}{2}t^2 - j2\pi\frac{B}{2}t + j\phi_0\right)$

bit = '1' $s_1(t) = \exp\left(j2\pi\frac{\alpha}{2}t^2 + j\phi_0\right)$

At RX:

$$y(t) \times s_0^*(t) = y(t) \exp\left(-j2\pi\frac{\alpha}{2}t^2 + j2\pi\frac{B}{2}t\right)$$

bit = '0' $= \exp(j\phi_0)$

bit = '1' $= \exp\left(j2\pi\frac{B}{2}t + j\phi_0\right)$



Pulse at $f = 0$

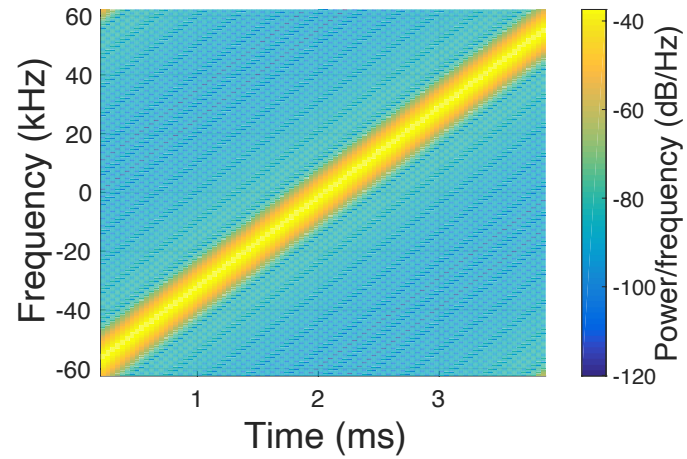
Pulse at $f = \frac{B}{2}$

But, How do we decode?

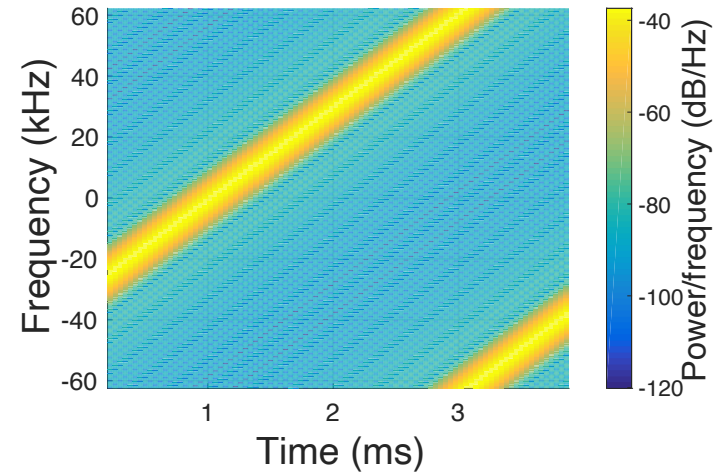
- (1) Demodulate by multiplying with $s_0^*(t)$
- (2) Take and FFT over symbol time
- (3) Find position of the peak

But, How do we decode?

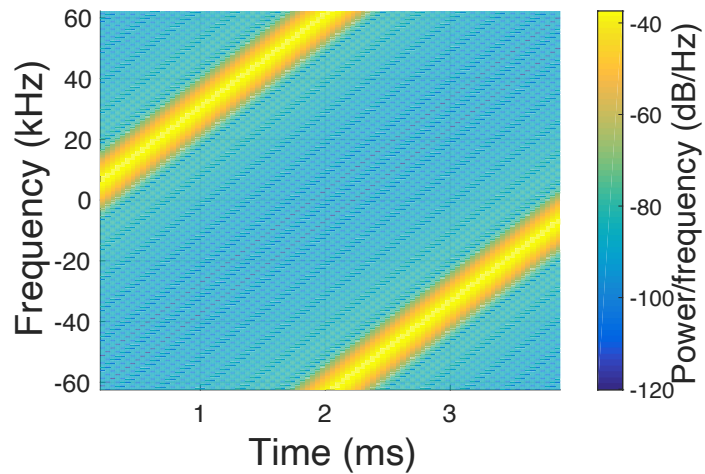
bits = '00'



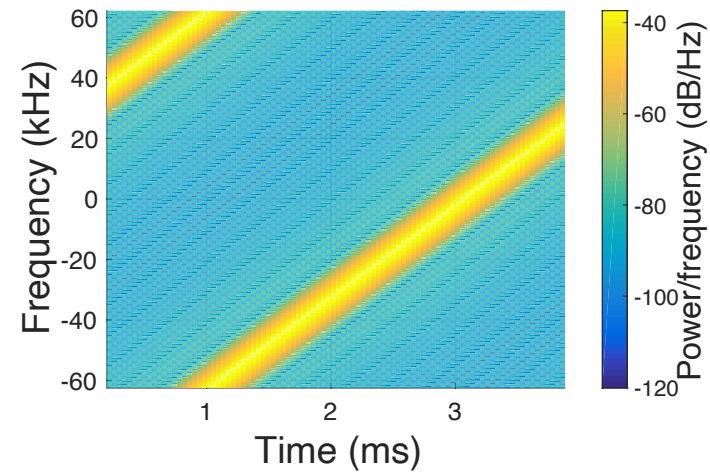
bits = '01'



bits = '10'

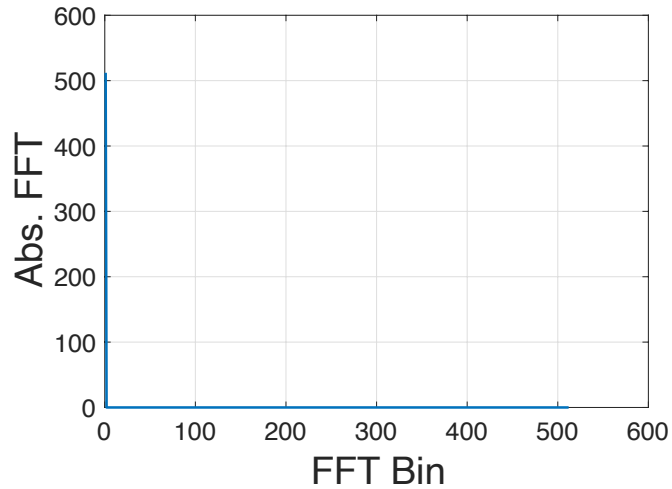


bits = '11'

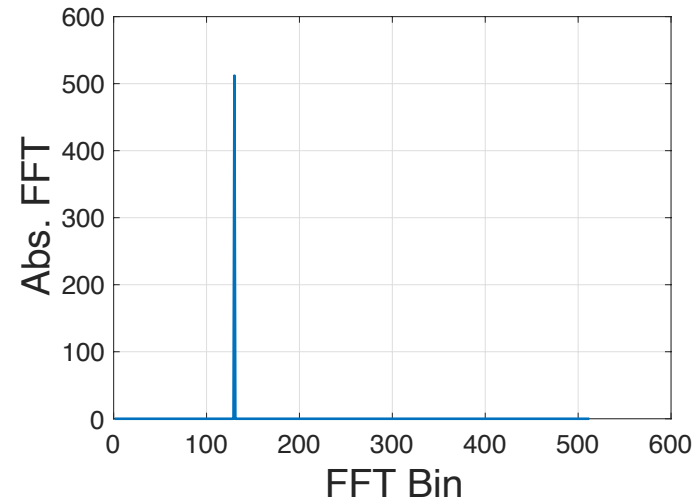


But, How do we decode?

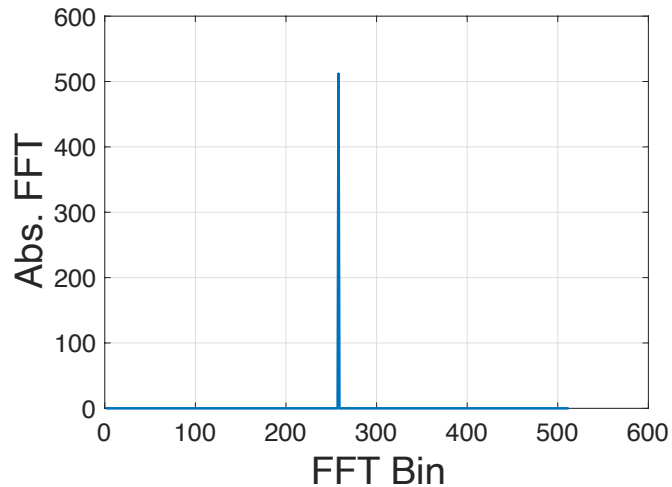
bits = '00'



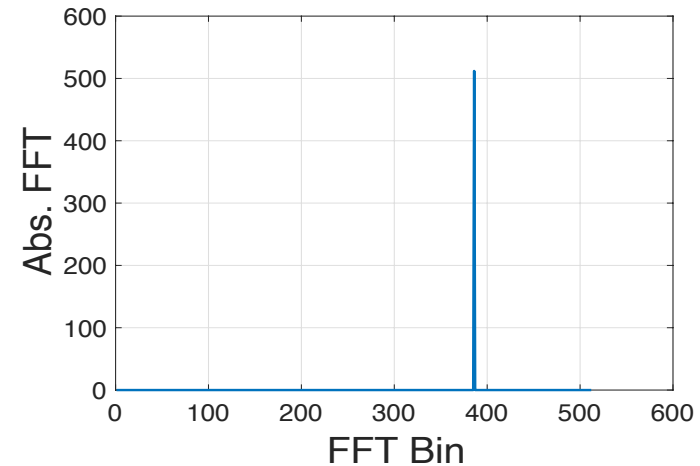
bits = '01'



bits = '10'



bits = '11'



What about the wireless channel?

LPWAN use low bandwidth (120kHz)

Narrow band channel

$$y(t) = h s_0(t) \times s_0^*(t)$$

$$\text{bit} = \text{'0'} = h \exp(j\phi)$$

$$\text{bit} = \text{'1'} = h \exp\left(j2\pi \frac{B}{2} t + j\phi\right)$$



Pulse at $f = 0$

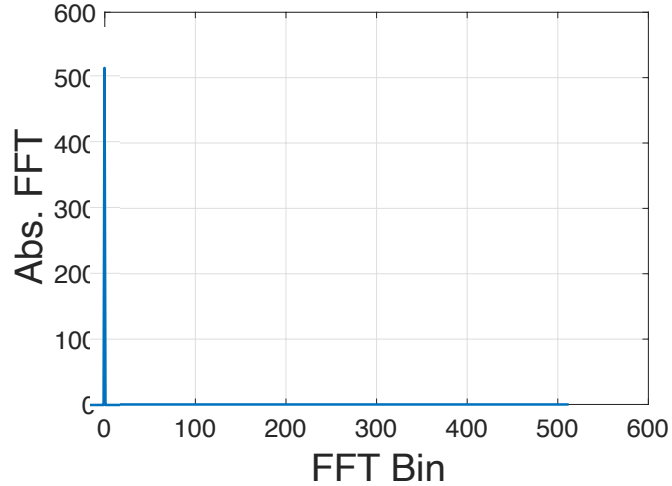
Pulse at $f = \frac{B}{2}$

Channel changes amplitude & phase but not
position of pulse

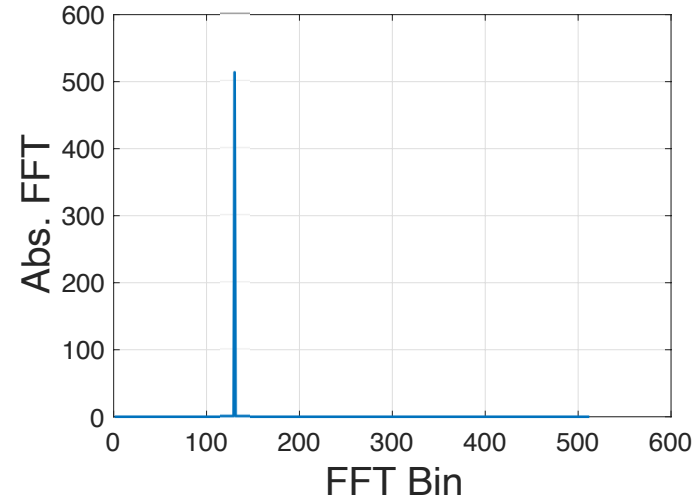
→ Can decode without correcting for channel

What about CFO & Sampling offset?

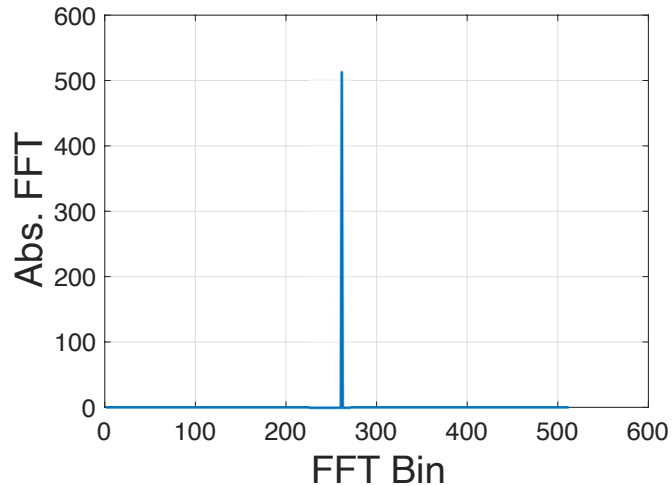
bits = '00'



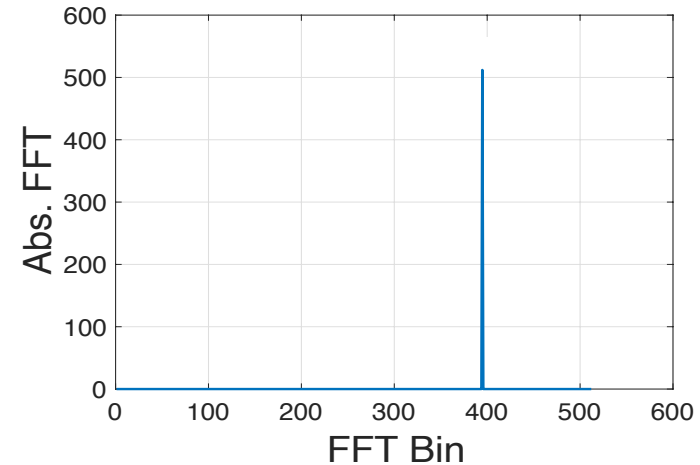
bits = '01'



bits = '10'

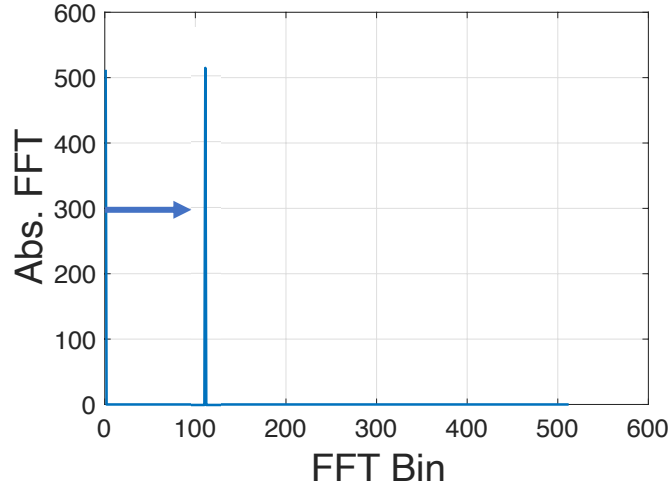


bits = '11'

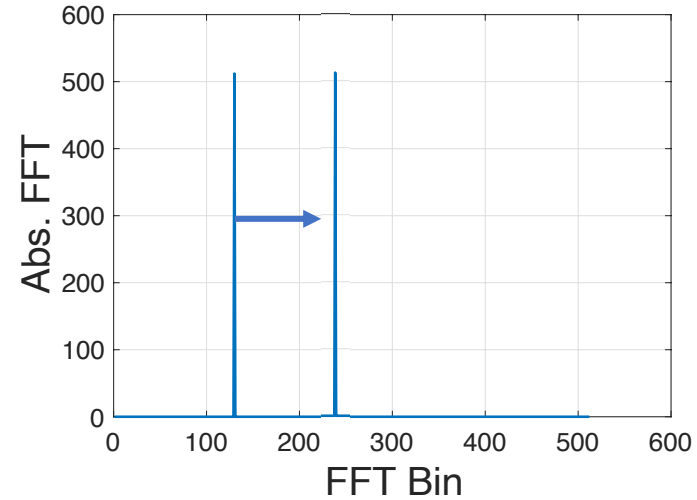


What about CFO & Sampling offset?

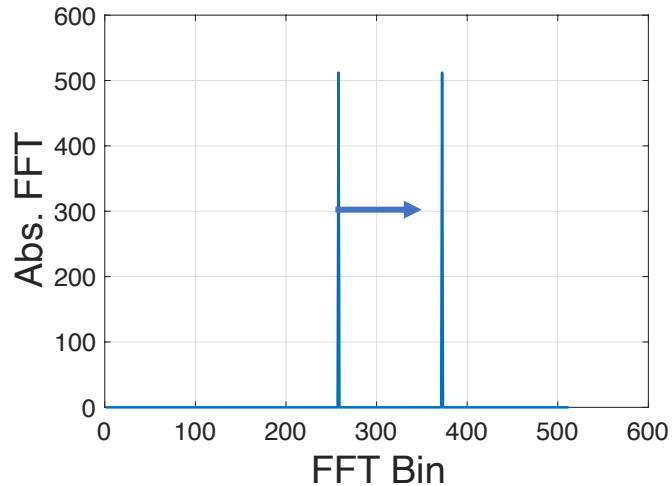
bits = '00'



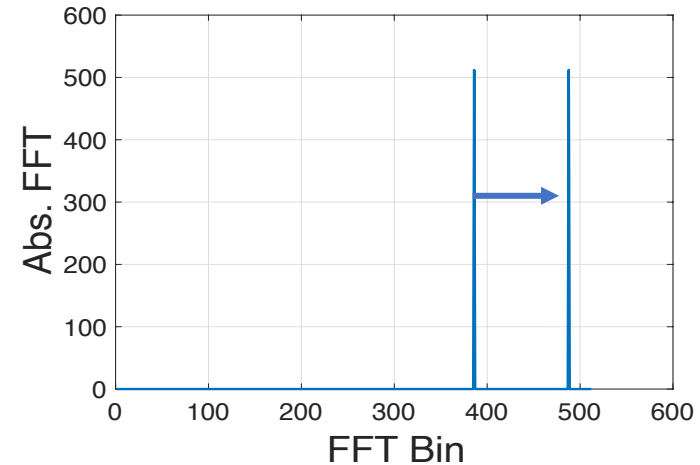
bits = '01'



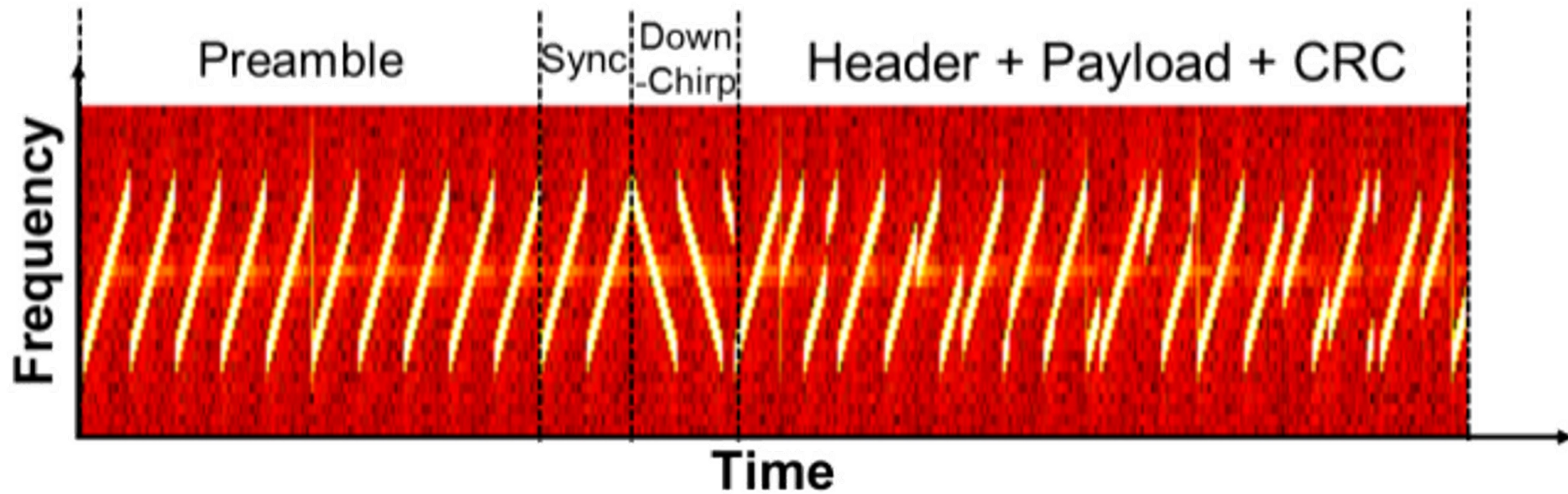
bits = '10'



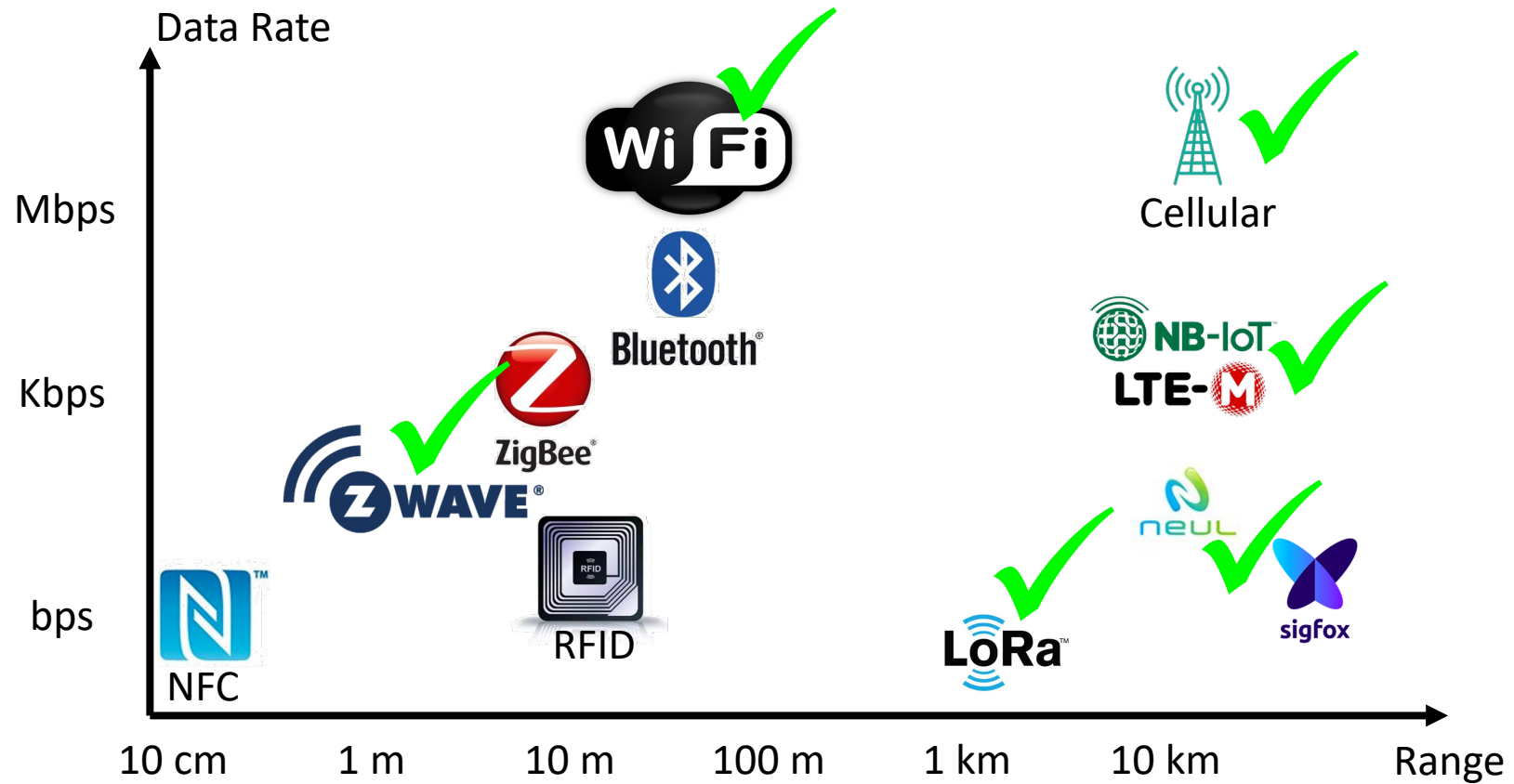
bits = '11'



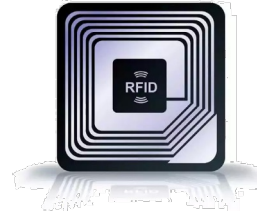
LoRaWAN™: Packet Structure



IoT Technologies



IoT: Backscatter Communication



- Low power: No Battery
- Low cost: 10 cents
- Low range: 10 – 15 meters
- Low Data rates: 10Kbps – 640 Kbps

RFID: Radio Frequency IDentification

Active RFID



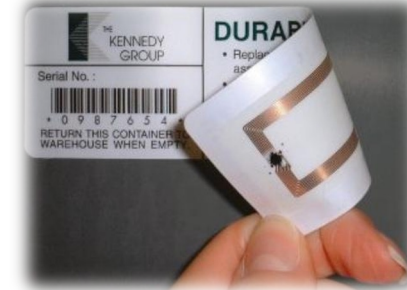
- Has battery
- Longer range
- Shorter life span
- Transmits its own signal using OOK

Battery Assisted RFID



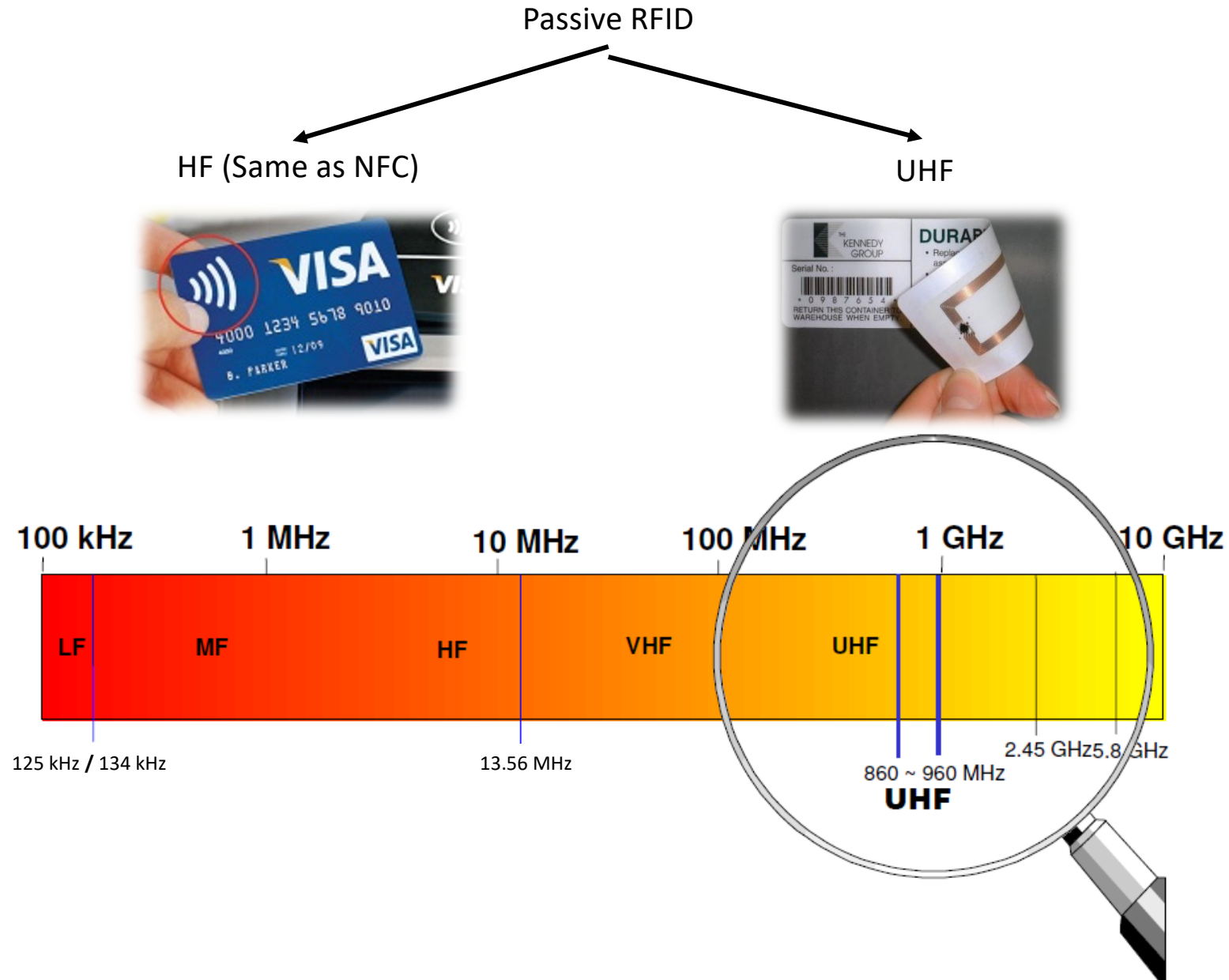
- Has battery
- Battery used from computation & sensing but not communication
- Backscatters a reader's signal using OOK

Passive RFID

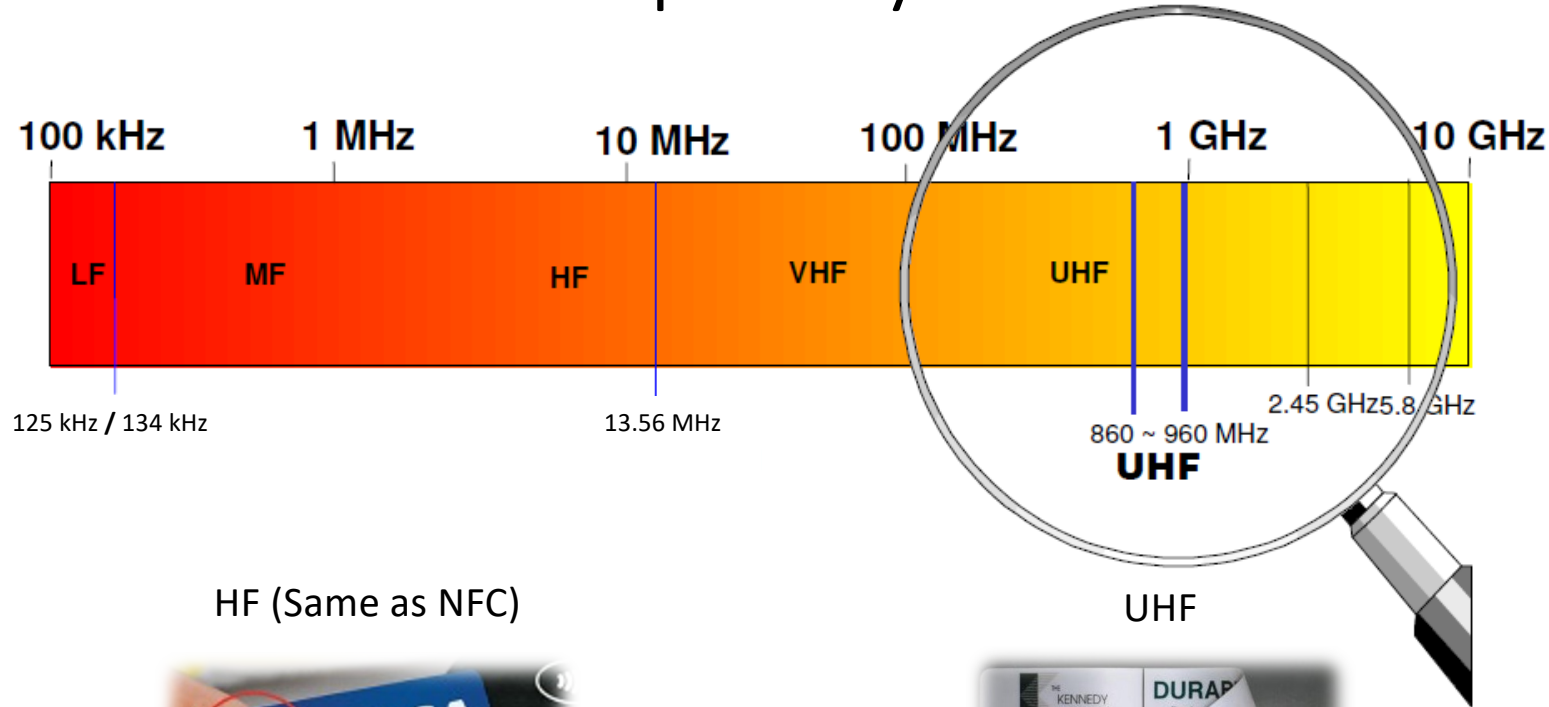


- No battery
- Short range
- Long life span
- Backscatters a reader's signal using OOK

RFID: Radio Frequency IDentification



RFID: Radio Frequency Identification



HF (Same as NFC)



Range: < 1cm

Data Rate: bps to few kbps

Technology: Backscatter over Inductive Coupling

UHF



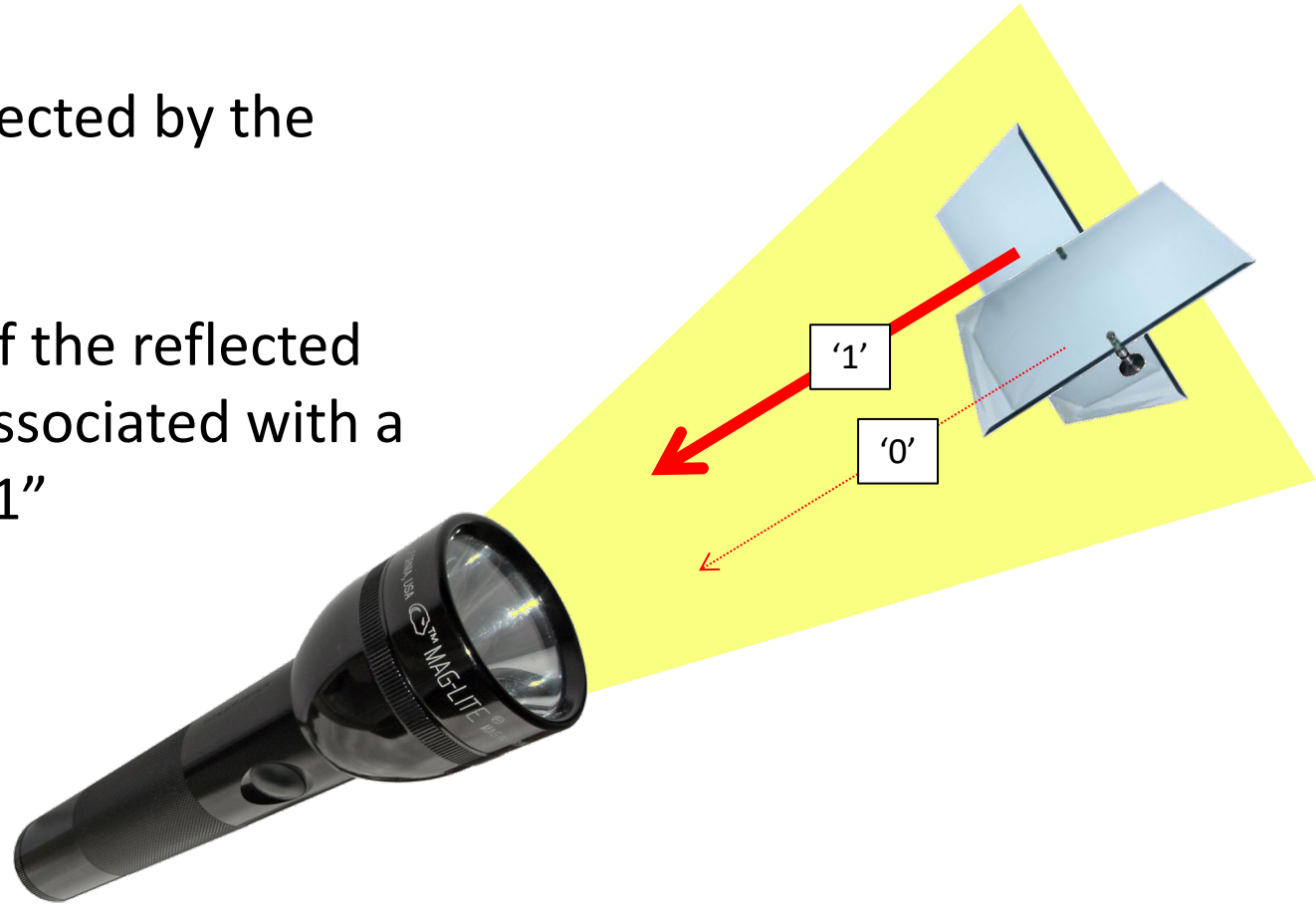
Few meters

100s kbps

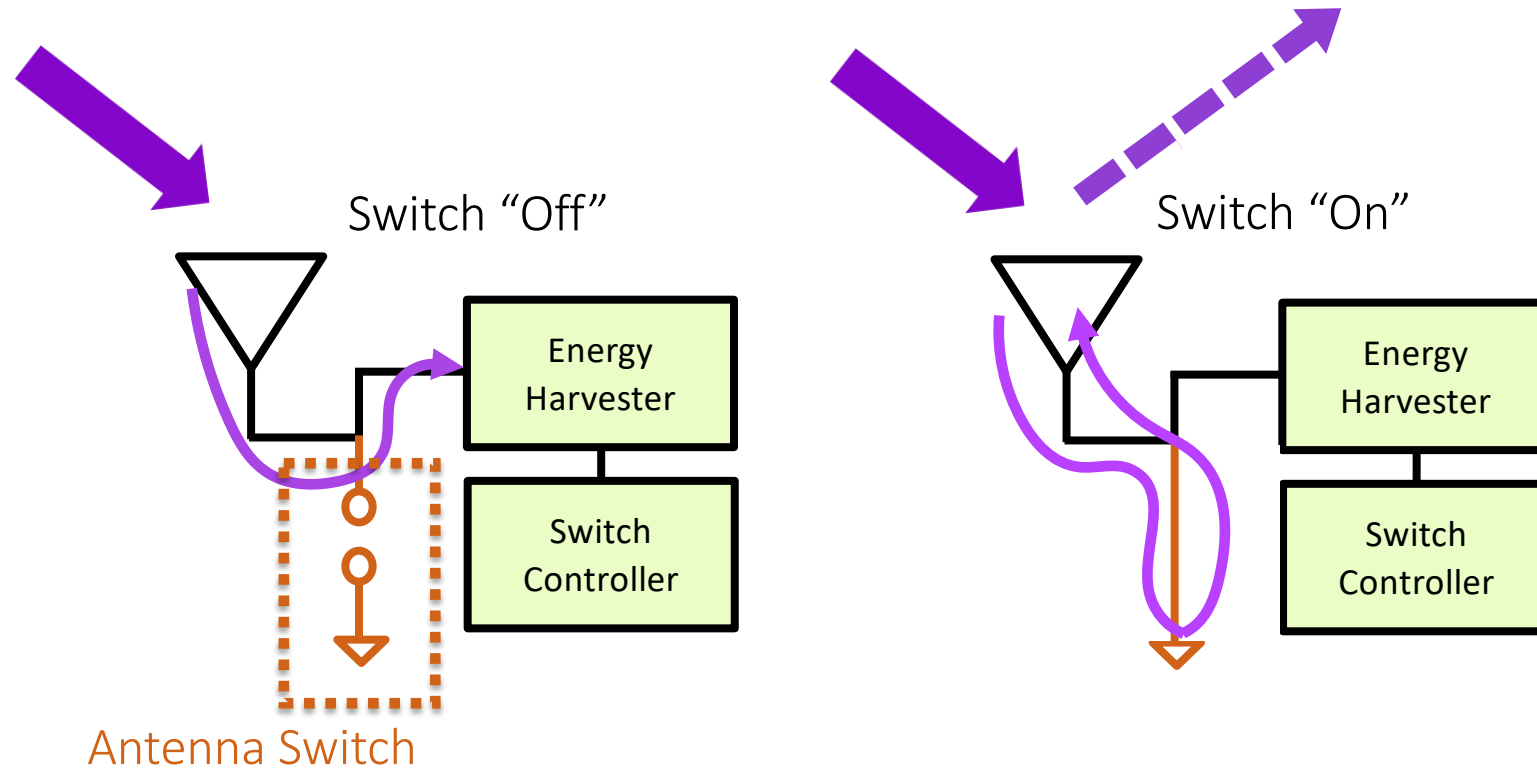
Backscatter over RF

Backscatter Communication

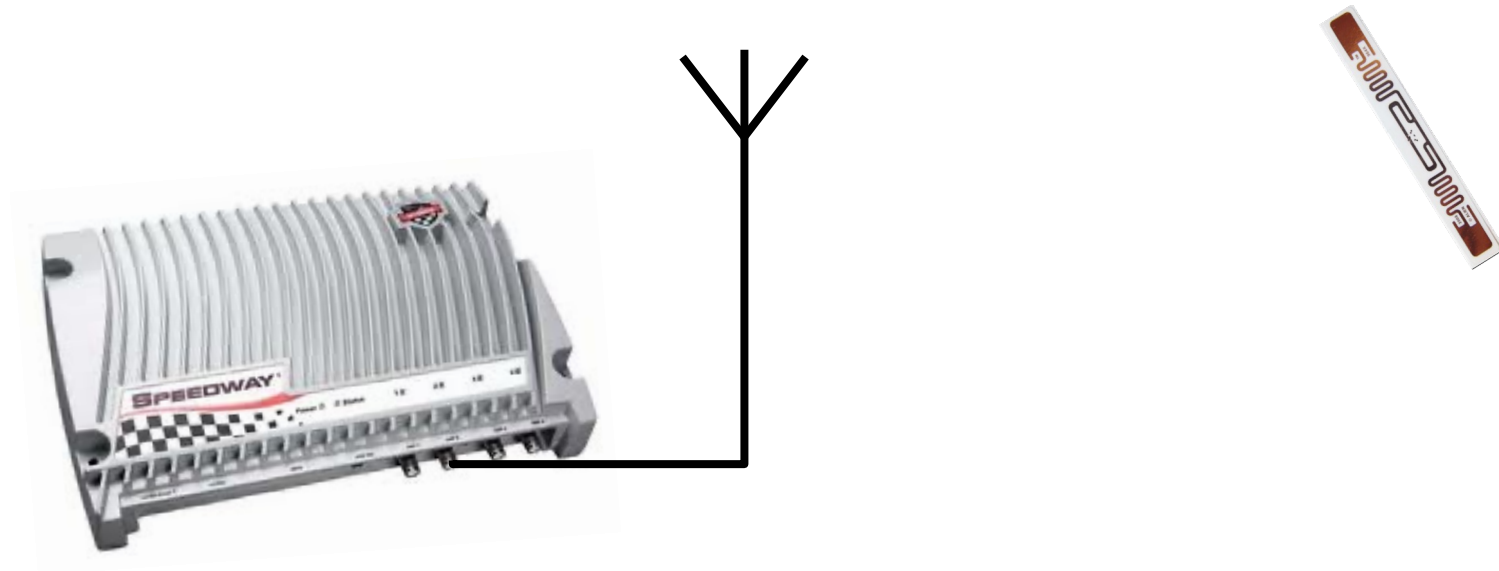
- A flashlight emits a beam of light
- The light is reflected by the mirror
- The intensity of the reflected beam can be associated with a logical “0” or “1”



Backscatter Communication



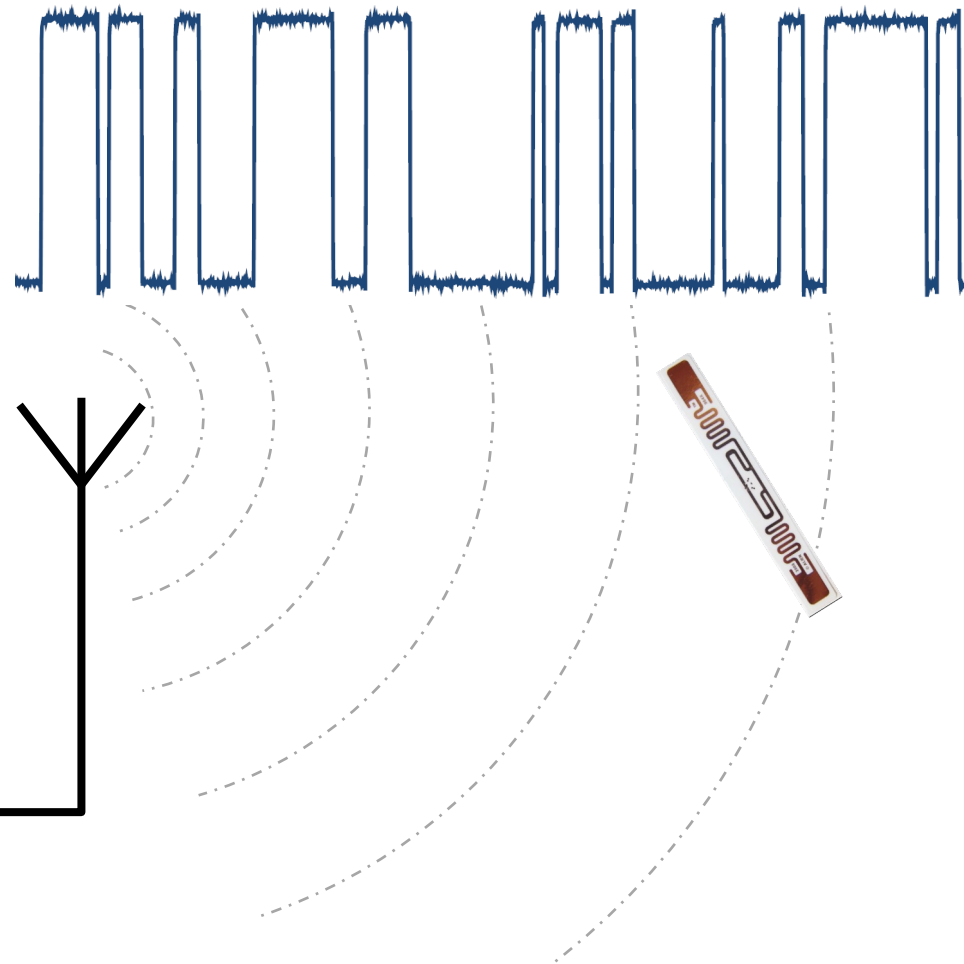
Backscatter Communication



Backscatter Communication

Tag reflects the reader's signal using ON-OFF keying

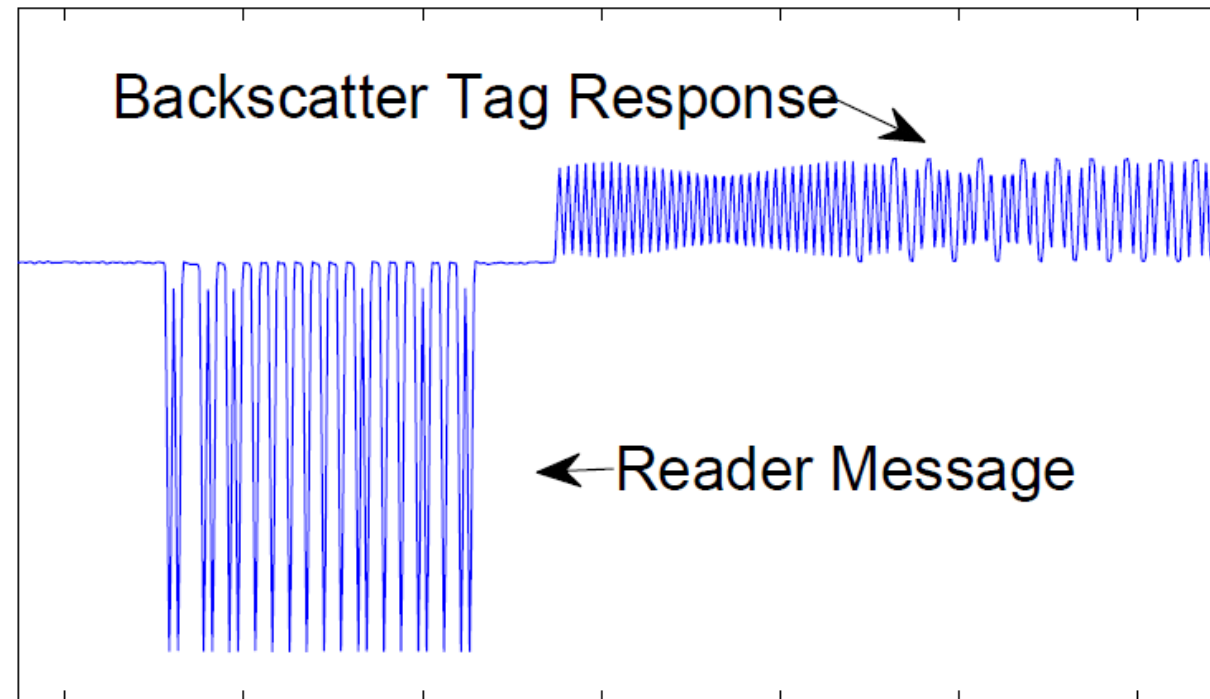
Reader shines an RF signal on nearby RFIDs



Backscatter Communication

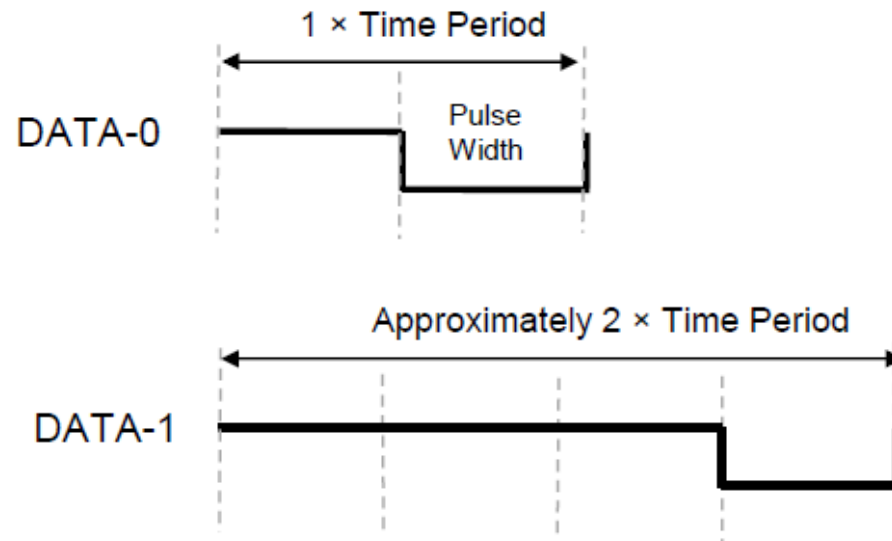
- Reader Transmits Continuous Sine Wave: $x(t) = \cos(2\pi f_c t)$
- Tag either reflect or doesn't reflect the signal

$$s(t) = \begin{cases} \alpha \cos(2\pi f_c t) & \text{bit} = 1 \\ 0 & \text{bit} = 0 \end{cases}$$



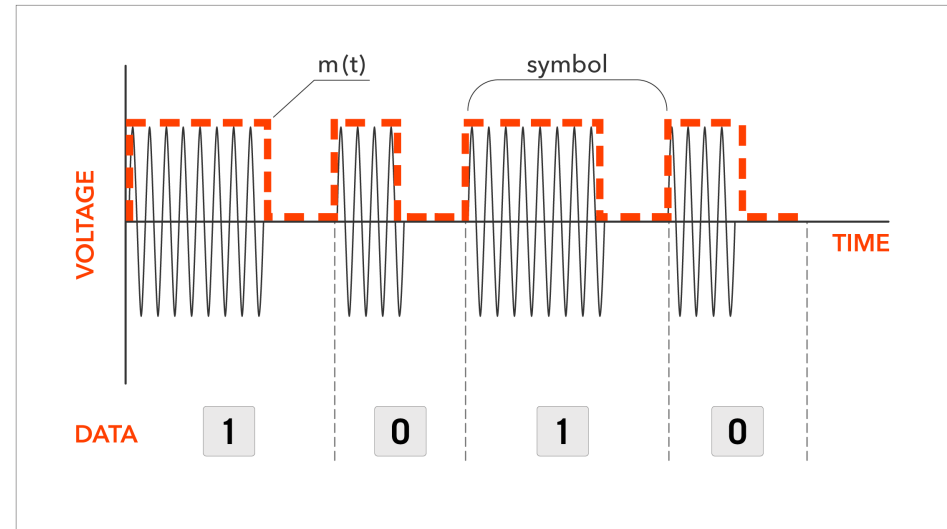
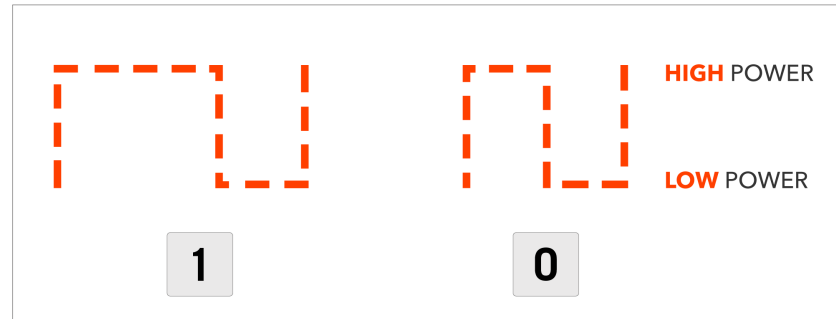
Backscatter Communication

- Both Reader and Tag Use ON-OFF Keying for modulation
- Bit Encoding, however, can differ.
- Reader-to-Tag Encoding: Pulse Interval Encoding (PIE)



Backscatter Communication

- Reader-to-Tag Encoding: Pulse Interval Encoding (PIE)

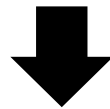


Backscatter Communication

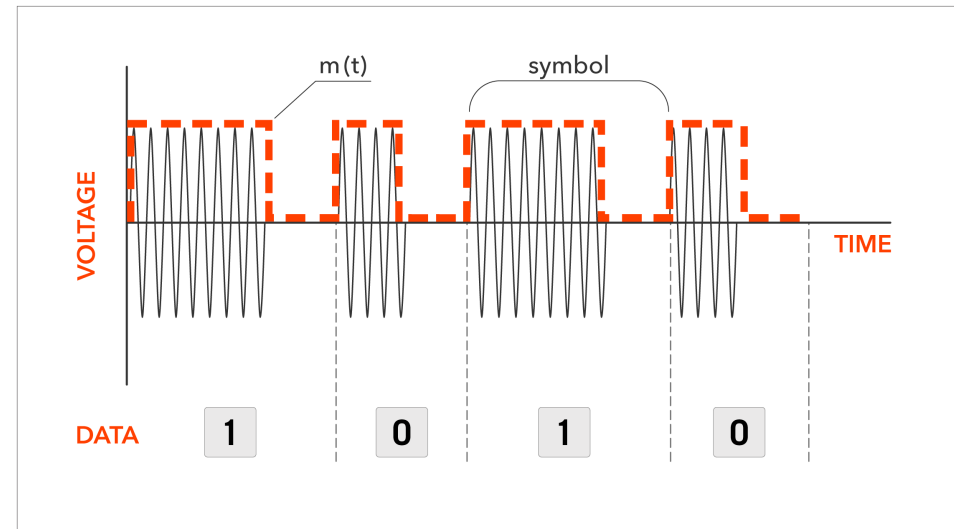
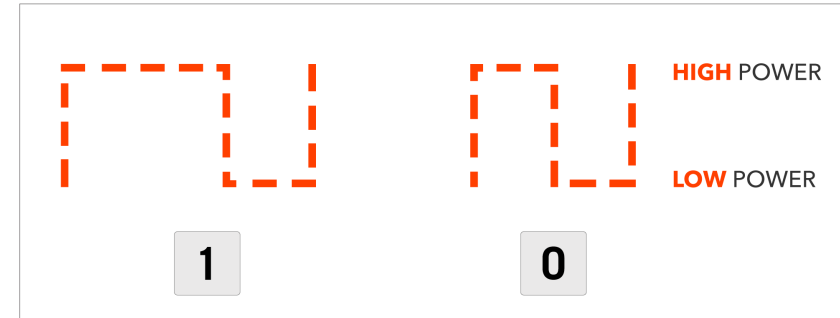
- Reader-to-Tag Encoding: Pulse Interval Encoding (PIE)

Why use PIE
encoding?

Signal is on for longer
time



Maximize energy
harvesting at the tag.

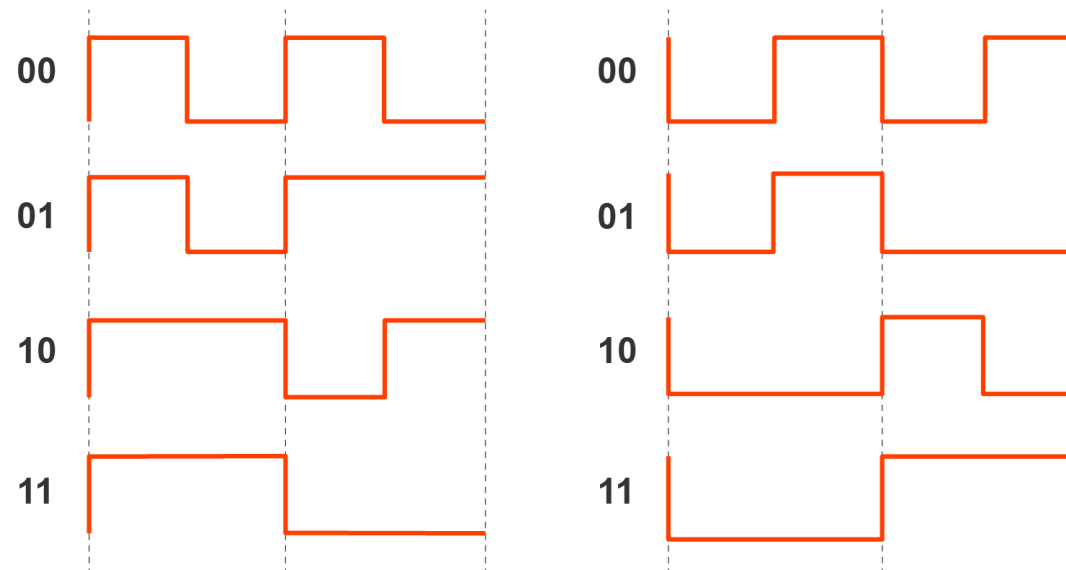
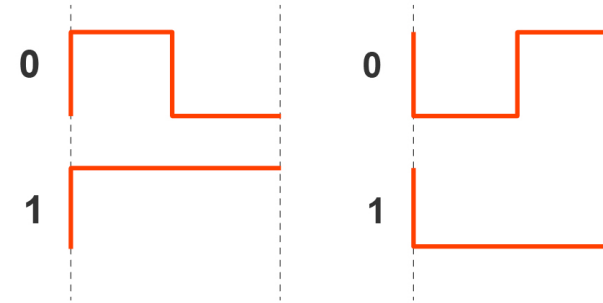


Backscatter Communication

- Tag-to-Reader Encoding:
 - FM0
 - Miller Code ($M=2, 4, 8$)

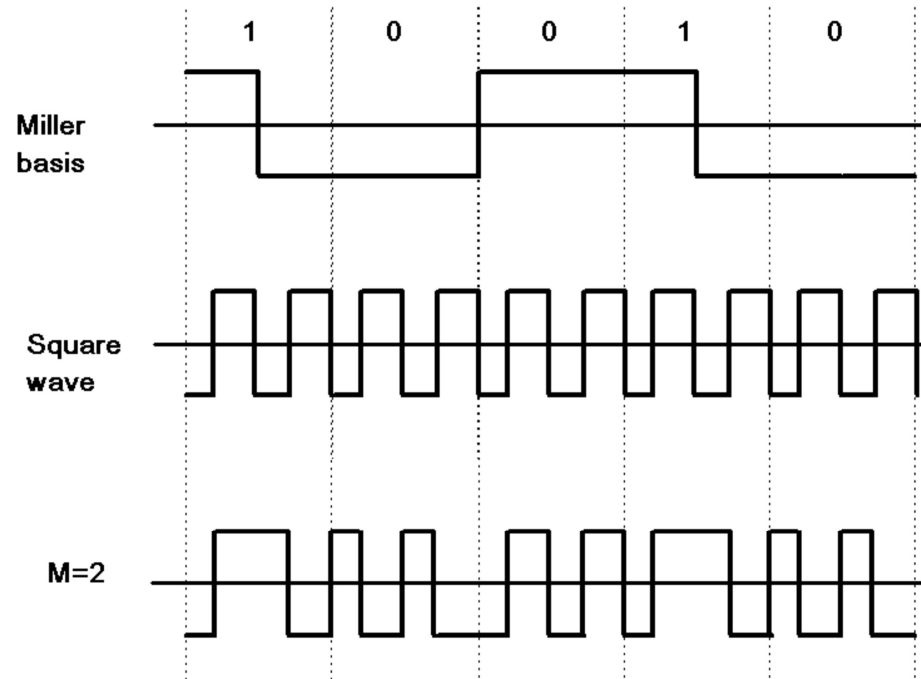
Backscatter Communication

- Tag-to-Reader Encoding: FM0
- Inverts the switch at every symbol
- 0 bits has extra switch mid-symbol



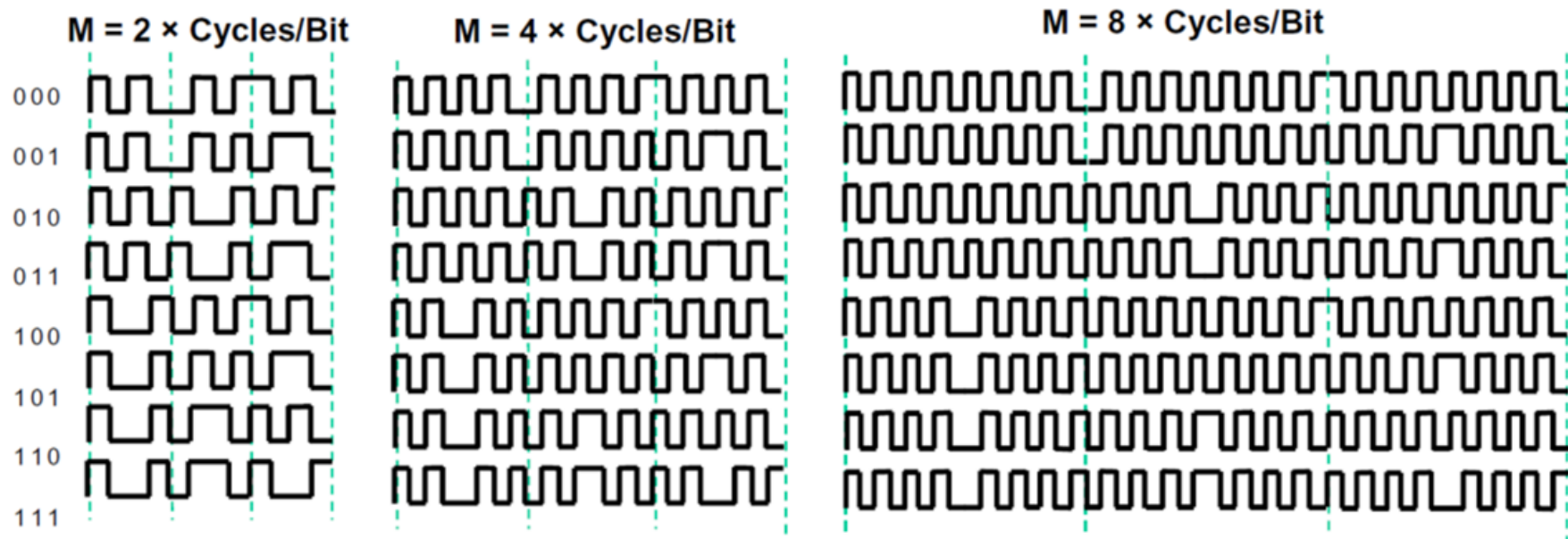
Backscatter Communication

- Tag-to-Reader Encoding: Miller
- Inverts the switch between two consecutive 0 bit symbols
- Inverts the switch in the middle of 1 bit symbol
- Multiple by square wave of M times symbol rate for M=2,4,8



Backscatter Communication

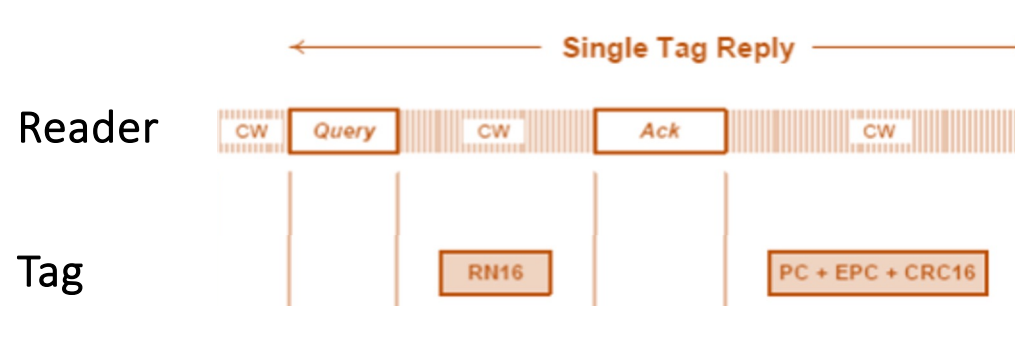
- Tag-to-Reader Encoding: Miller
- Inverts the switch between two consecutive 0 bit symbols
- Inverts the switch in the middle of 1 bit symbol
- Multiple by square wave of M times symbol rate for M=2,4,8



Backscatter Communication

- Tag-to-Reader Encoding:
 - FM0: High Data Rate: 40 Kbps- 640 Kbps
 - Miller Code (M=2, 4, 8)
 - Multiple switches per bit.
 - Robust to Multi-Reader, Multi-Tag scenarios.
 - Robust to noise.
 - M=2, Data Rate: 20 Kbps – 320 Kbps
 - M=4, Data Rate: 10 Kbps – 160 Kbps
 - M=8, Data Rate: 5 Kbps – 80 Kbps

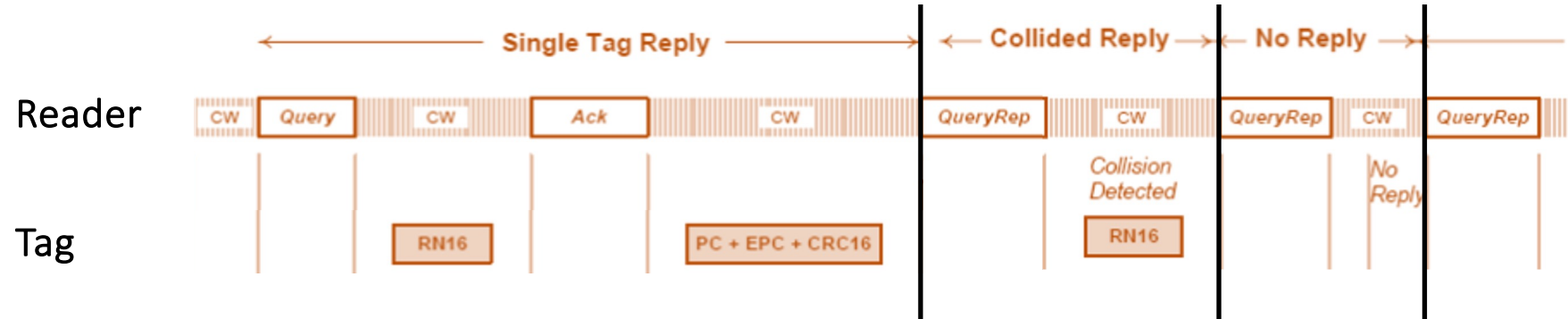
EPC Gen2 Standard – MAC



Slotted Aloha:

- Reader allocates Q time slots and transmits a query at the beginning of each time slot
- Each tag picks a random slot and transmits a 16-bit random number
- In each slot:
 - RN16 decoded → Reader ACKs → Tags transmits 96-bit ID
 - Collision → Reader moves on to next slot
 - No reply → Reader moves on to next slot

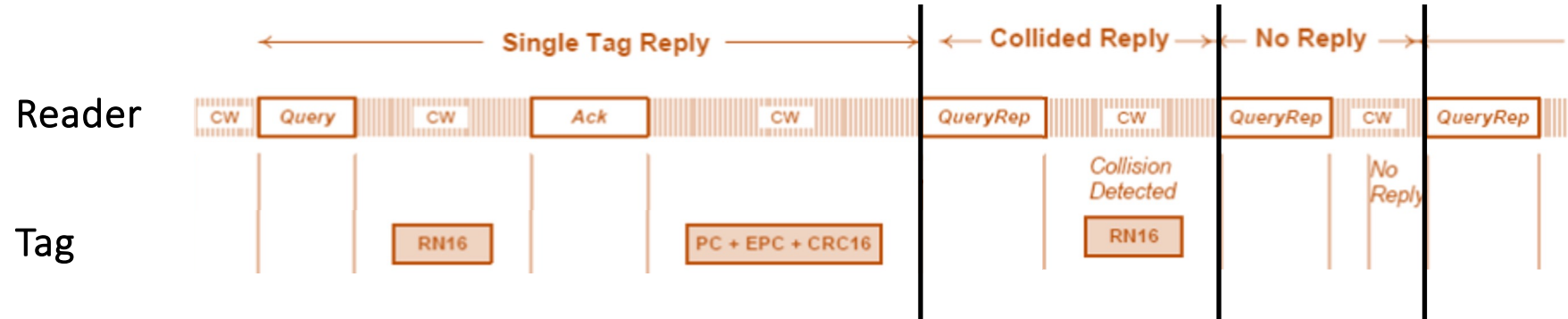
EPC Gen2 – MAC



Inefficient:

- If reader allocates large number of slots → Too many empty slots
- If reader allocates small number of slots → Too many collisions

EPC Gen2 – MAC



Inefficient:

- If reader allocates large number of slots → Too many empty slots
- If reader allocates small number of slots → Too many collisions
- If reader knows number of tags = N → Allocate $K=N$ slots → 37% efficiency
- Downlink overhead