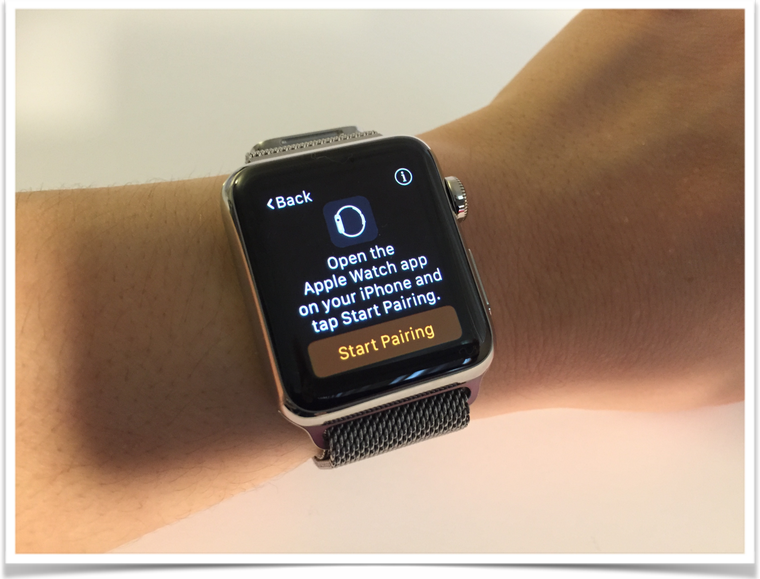
SyncVibe: Fast and Secure Device Pairing through Physical Vibration on Commodity Smartphones

Kyuin Lee[1], Vijay Raghunathan[2], Anand Raghunathan[2], and Younghyun Kim[1]

[1]Department of Electrical and Computer Engineering, University of Wisconsin—Madison
[2]School of Electrical and Computer Engineering, Purdue University

Challenges in Pervasive Computing

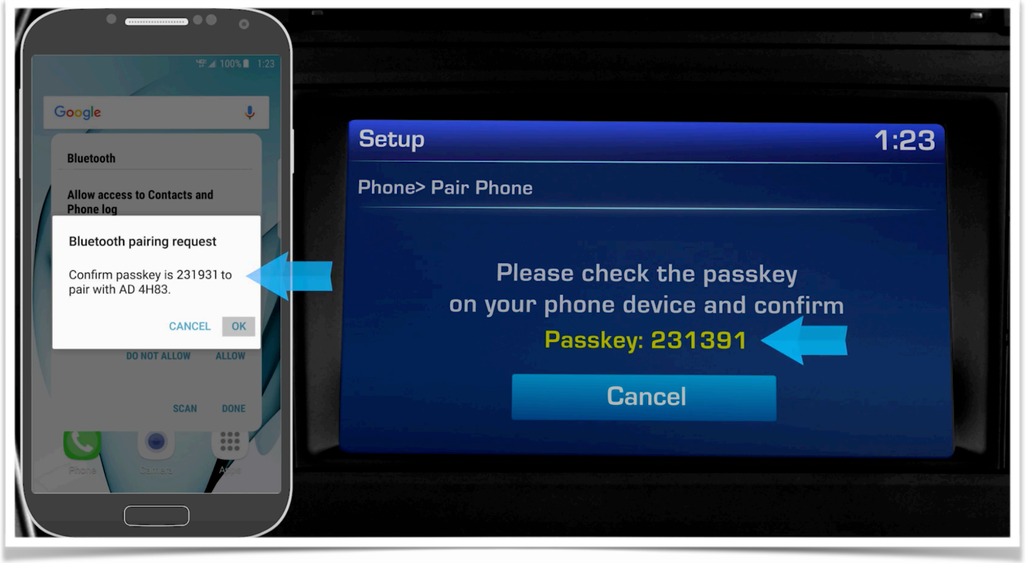
- Constant pairing and un-pairing between devices
- Current pairing paradigm degrades user experience
  - Manually entering pins
- Results in bulky user interface components
  - Screens, keyboards, etc.



Pairing Procedure of Bluetooth

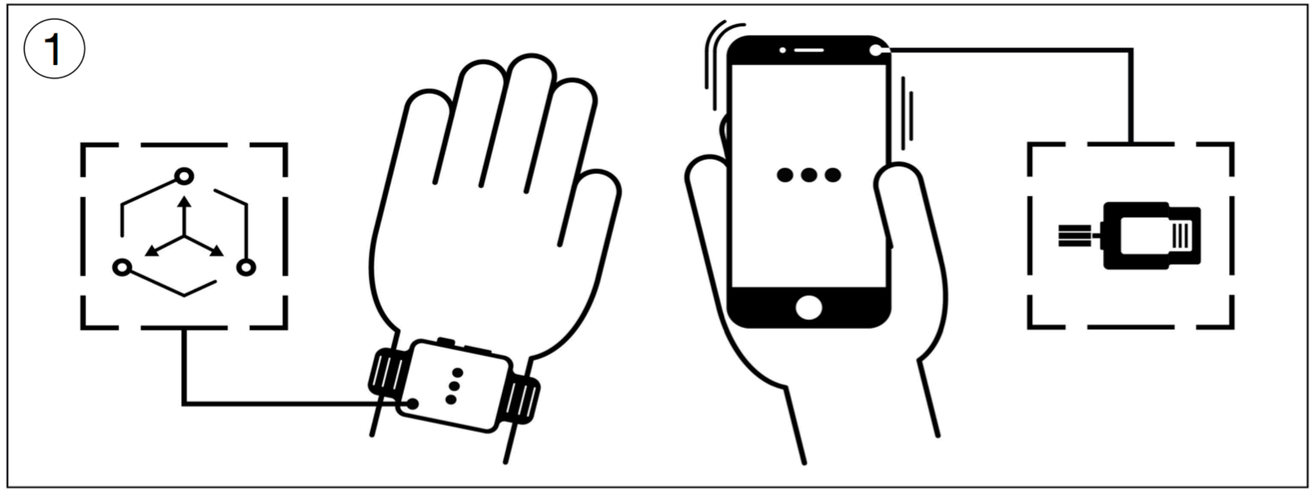
1. Turn Bluetooth on and search

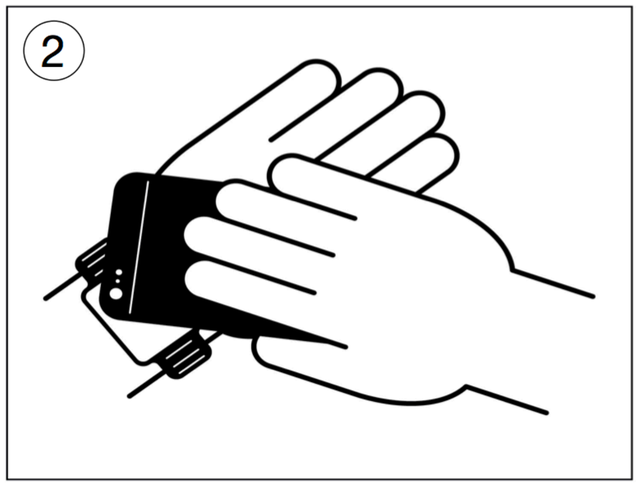
2. Passkey generated by the host

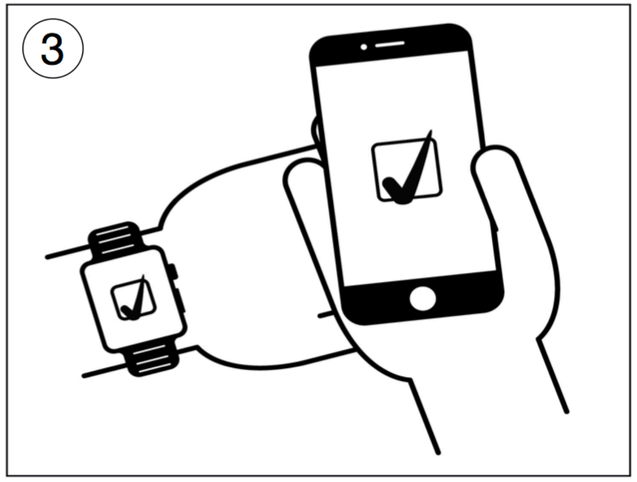


3. Confirm or enter the passkey

SyncVibe: Pairing through Physical Vibrations







1. Two pairing devices that share no prior knowledge

2. Direct contact between devices while motor vibrates

3. High-bandwidth wireless connection established

SyncVibe: Key Features

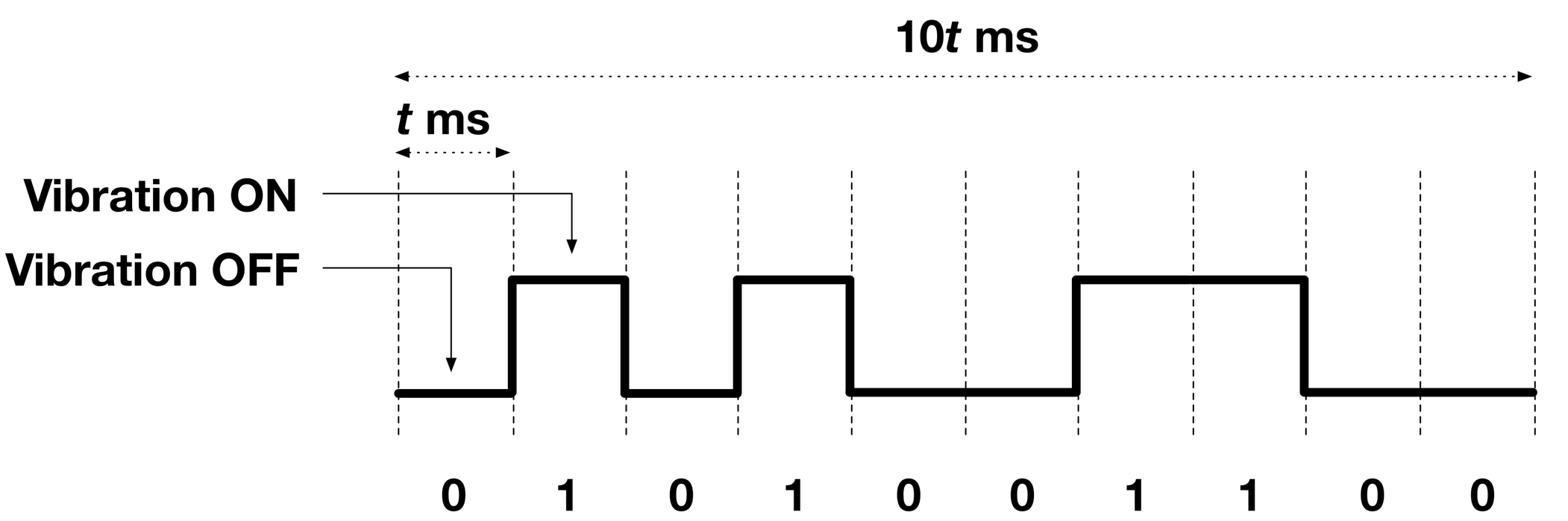
Usability — Simple and intuitive pairing procedure

Low Overhead — Small-footprint and low-cost hardware

Security — Protection against eavesdropping and man-in-the-middle attacks

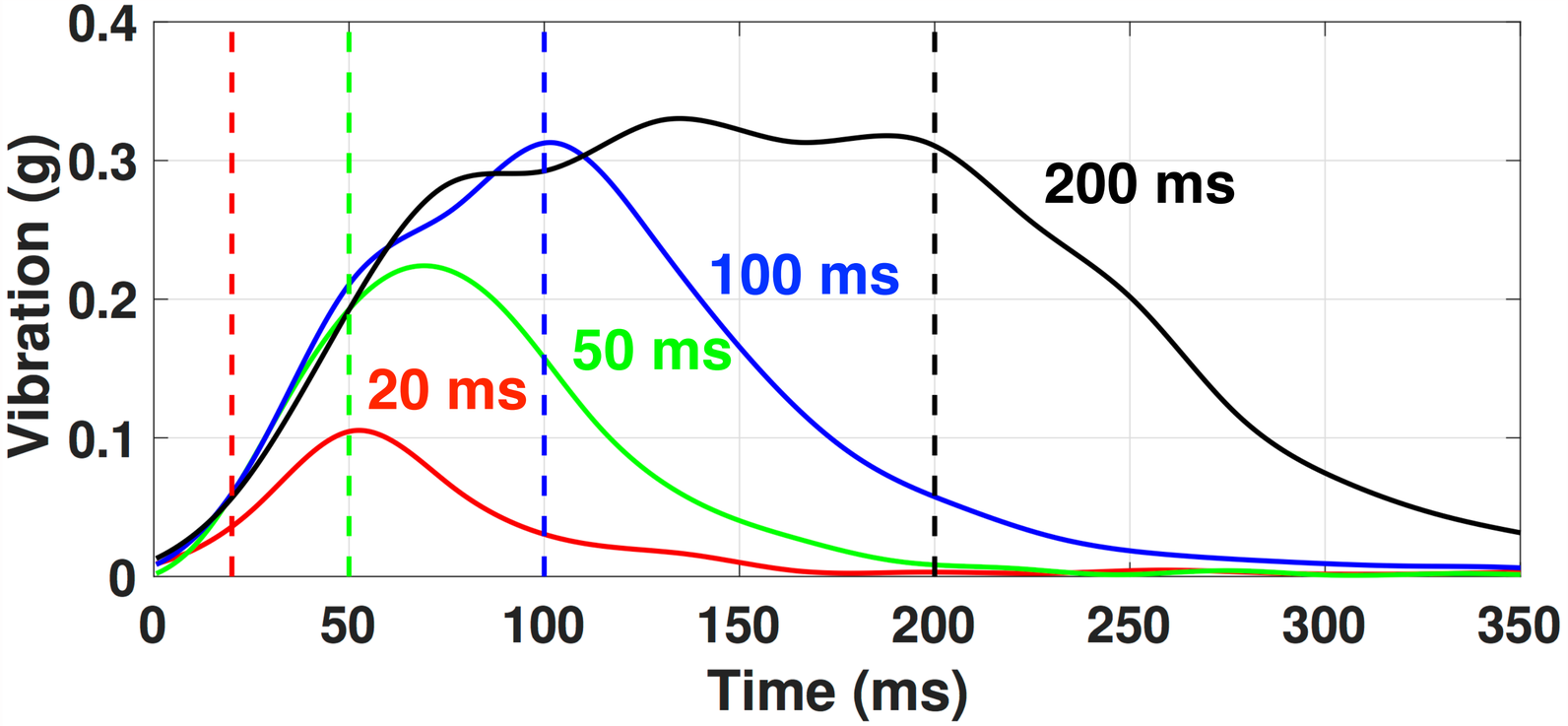
On-off Keying Modulation

- On-off keying (OOK) is commonly used method to modulate data
  - Vibration period (*t*) must be controlled with fine granularity



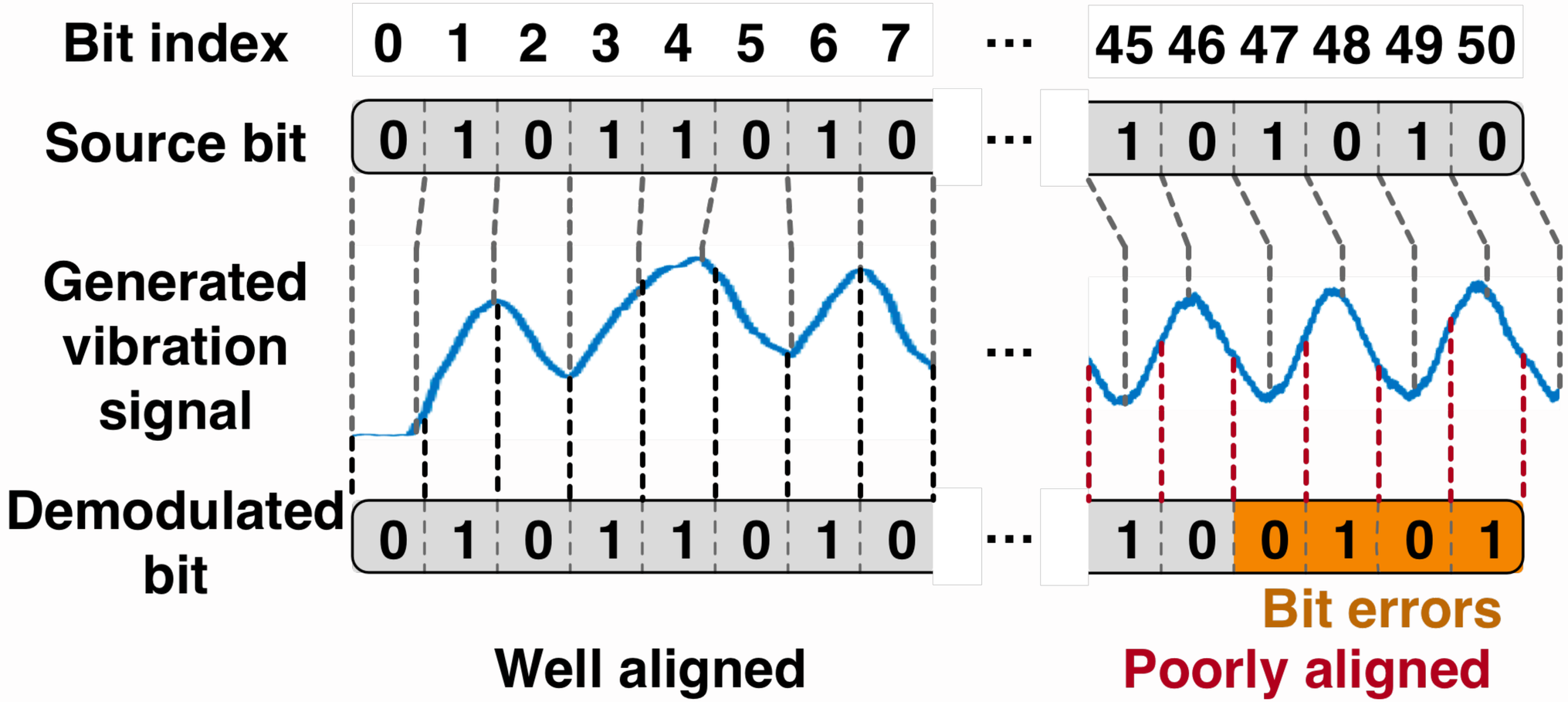
Challenges in Vibratory Communication

- Lack of precise timing control due to:
  - 1. Slow vibration motor response
  - 2. Non-real-time property of OS



Challenges in Vibratory Communication

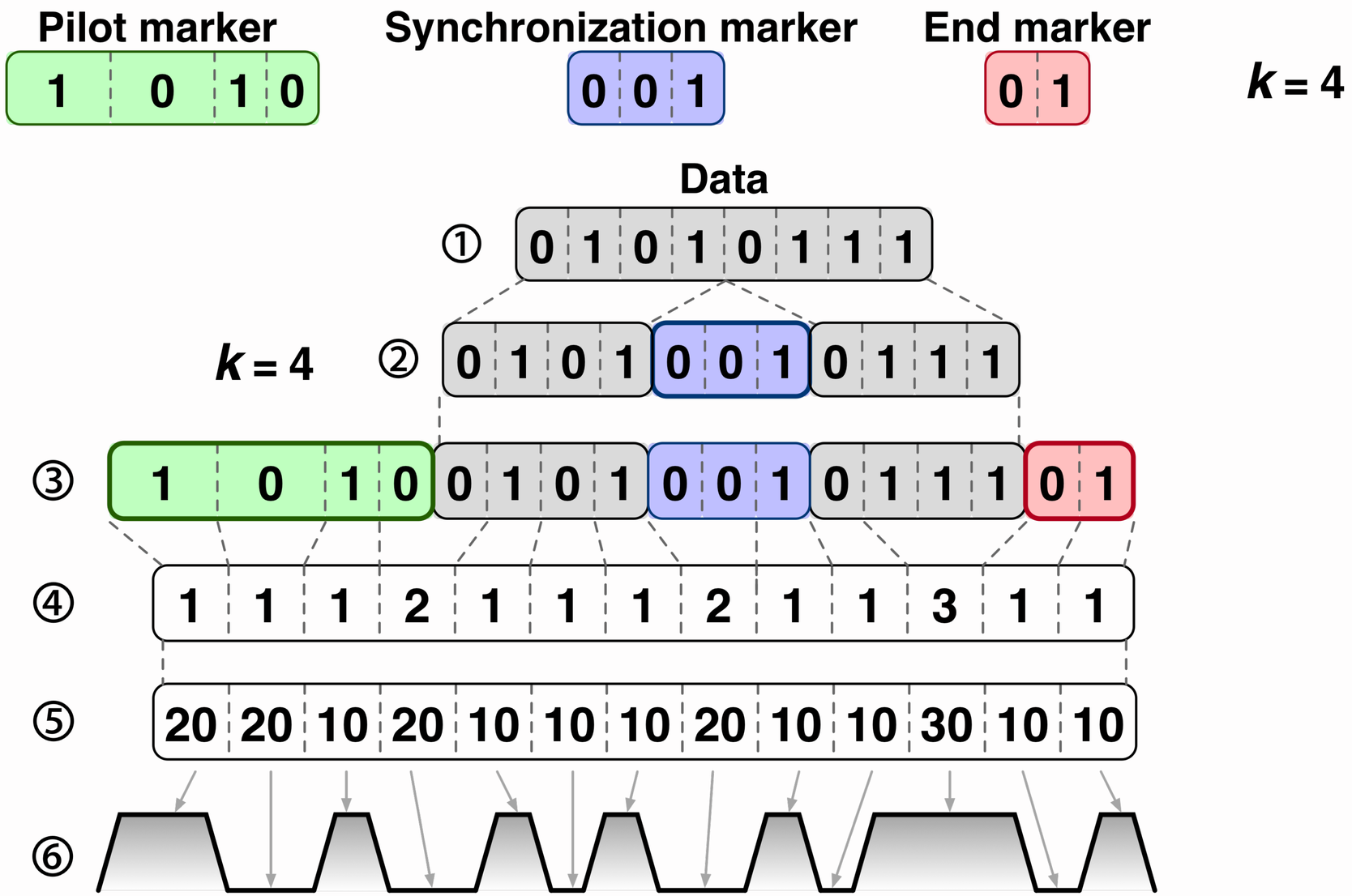
- Asynchronous transmission



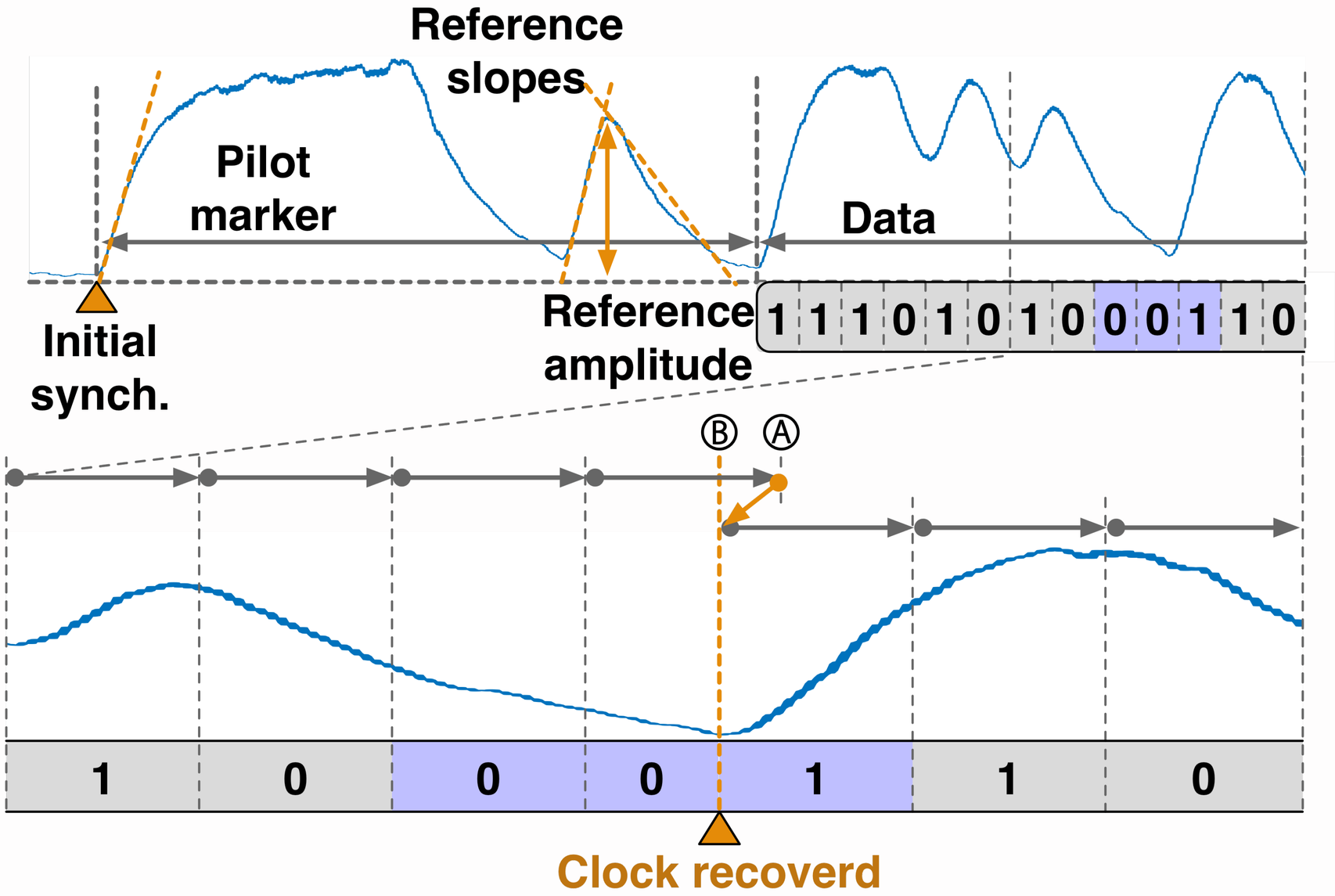
Synchronization

- SyncVibe uses OOK
- ***Synchronization pattern:*** **where synchronization takes place**
  - **Bit pattern of multiple 0s followed by 1 (i.e., 0001)**
  - **Inserted only when *k* consecutive bits do not contain synchronization pattern**
- ***Pilot marker*** **: signals start of data transmission**
  - **Allows receiver to measure maximum vibration amplitude**

SyncVibe - Modulation



SyncVibe - Demodulation



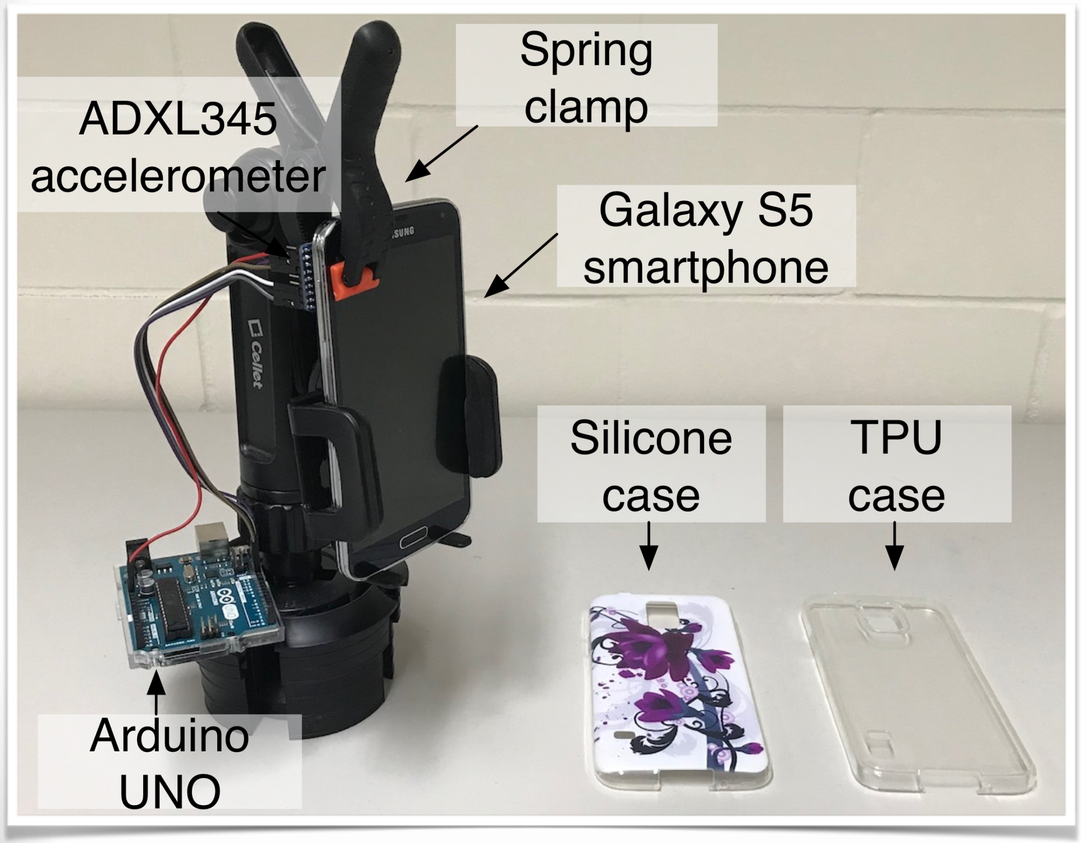
Expected Bits per Second

Expected bps = bps x effective bit ratio x success rate

$$= \frac{1}{t} \times \frac{l}{l+s} \times r \quad \text{(bps)}$$

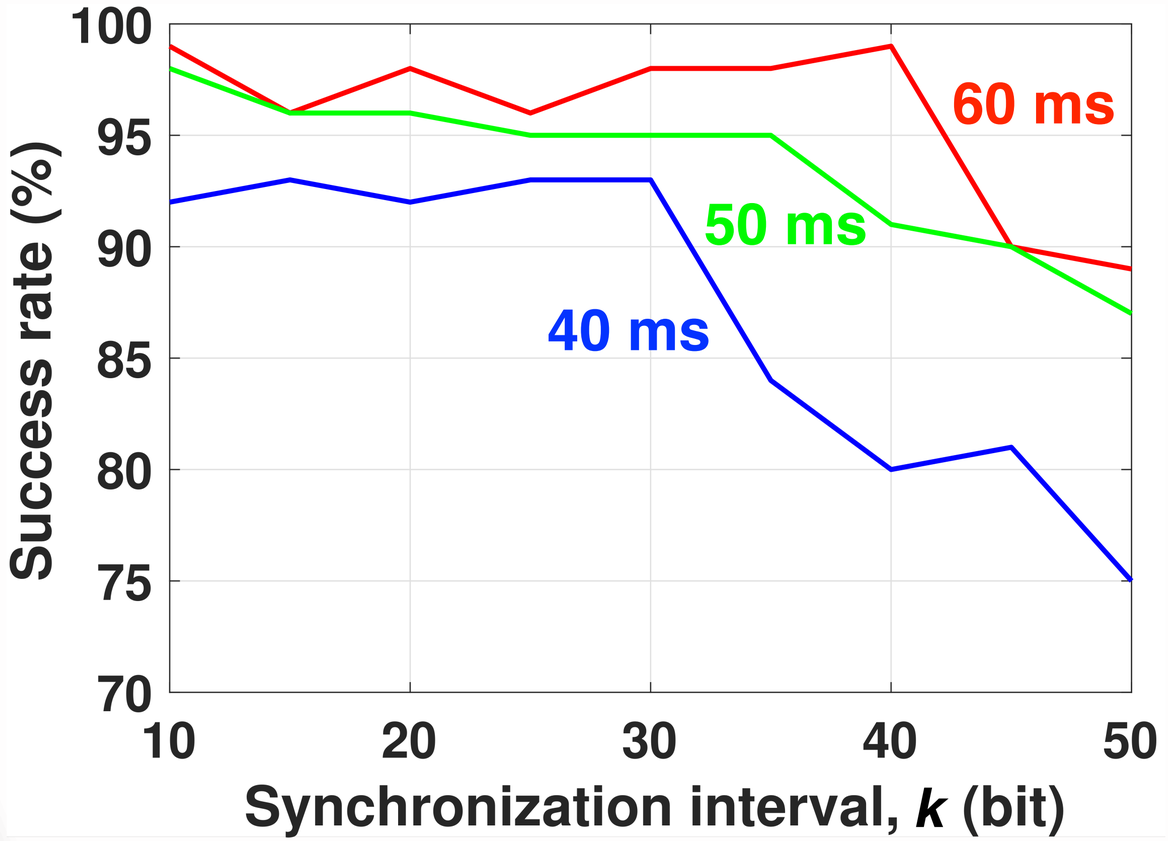
- ***r*** **: rate of successful pairing attempts**
- ***t*** **: time-interval representing each bit**
- ***l*** **: length of pairing data bits**
- ***s*** **: total number of overhead bits (pilot, end and synchronization markers) added**
- ***k*** **: length of maximum un-synchronizable bits**
- **Small *k* : relatively ↓ effective bit ratio, relatively ↑ success rate**
- **Large *k* : relatively ↑ effective bit ratio, relatively ↓ success rate**

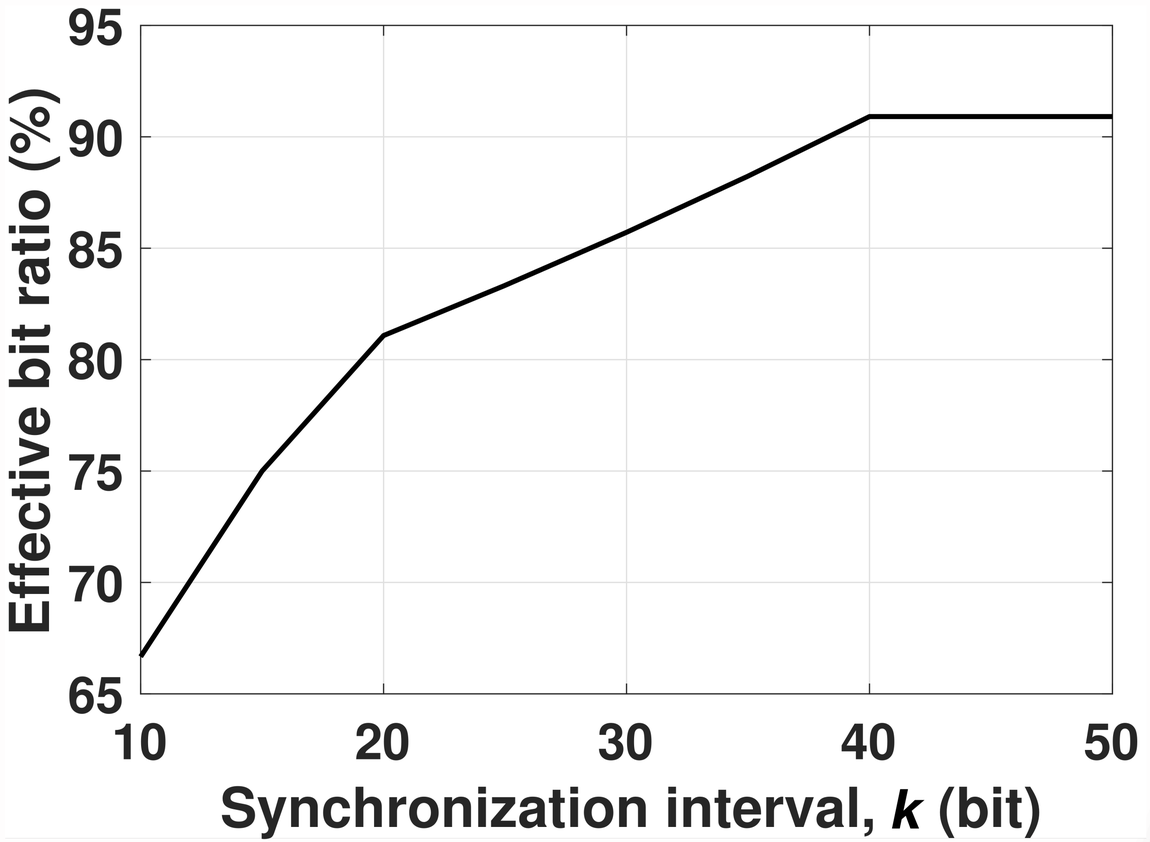
Prototype



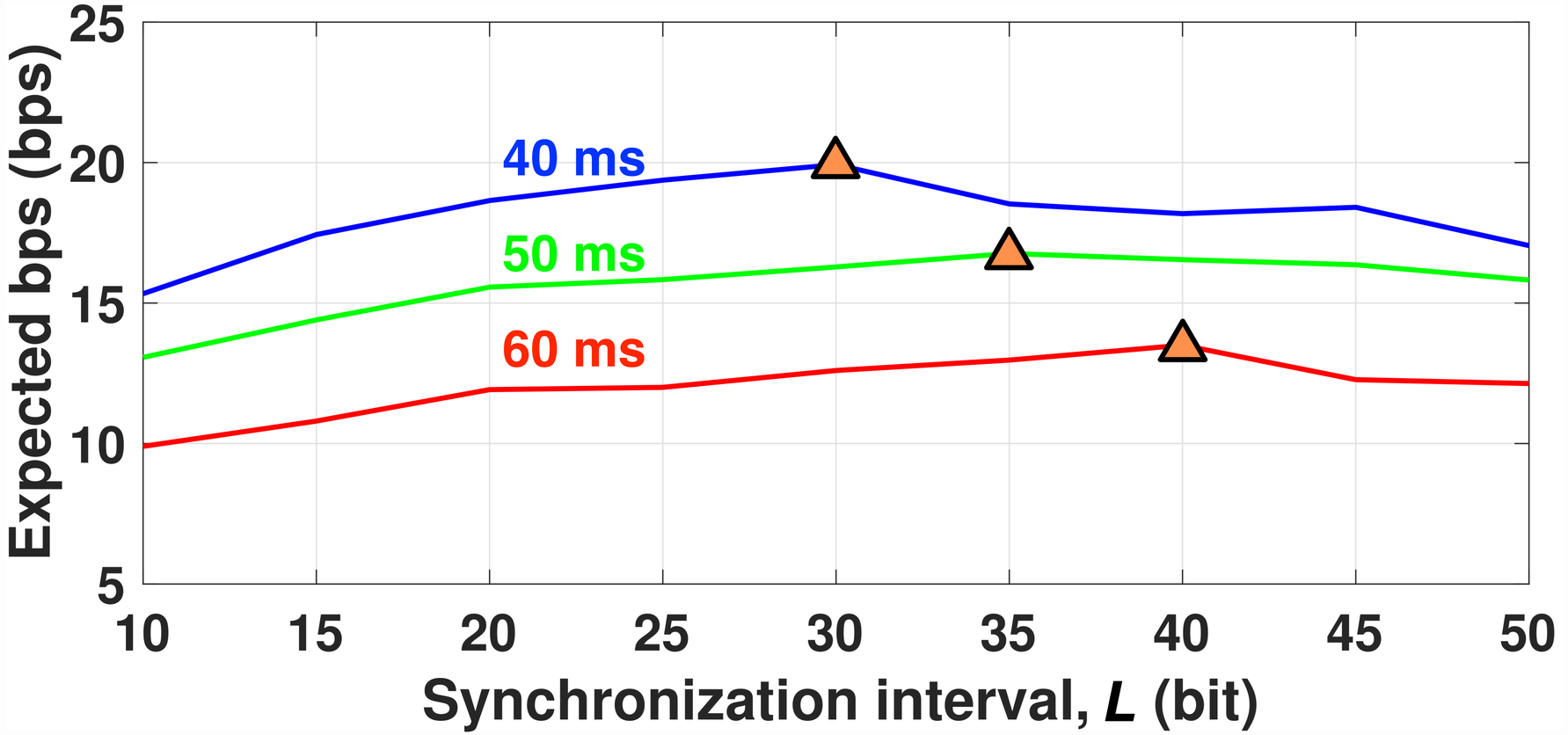
- **Standard Android API**
  - **No H/W and kernel modification**
- **ADXL345 at 1600 Hz**
  - **Most commonly used accelerometer**
- **Synchronization pattern: 00001**
- **Vibration period (*t*) = 40, 50, and 60 ms**
- **100 samples of 150 (*L*) data bits**

Trade-off in Expected BPS



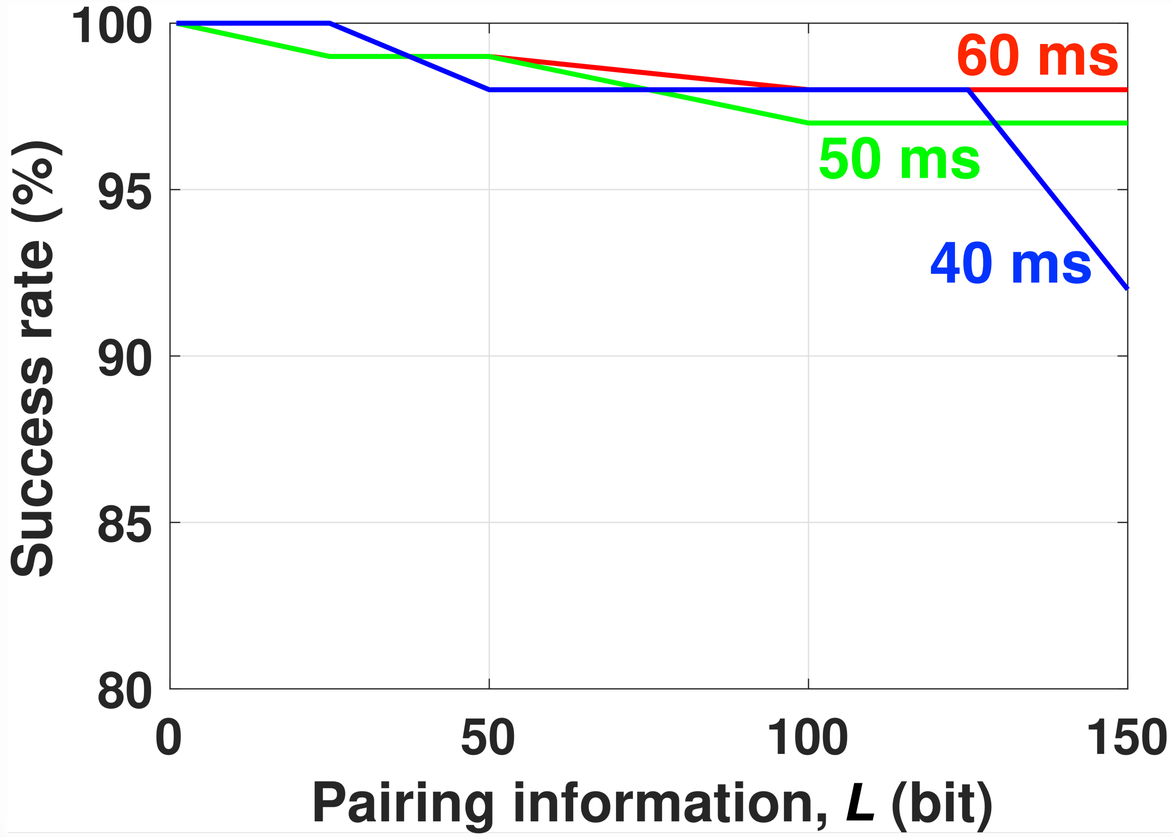


Trade-off in Expected BPS

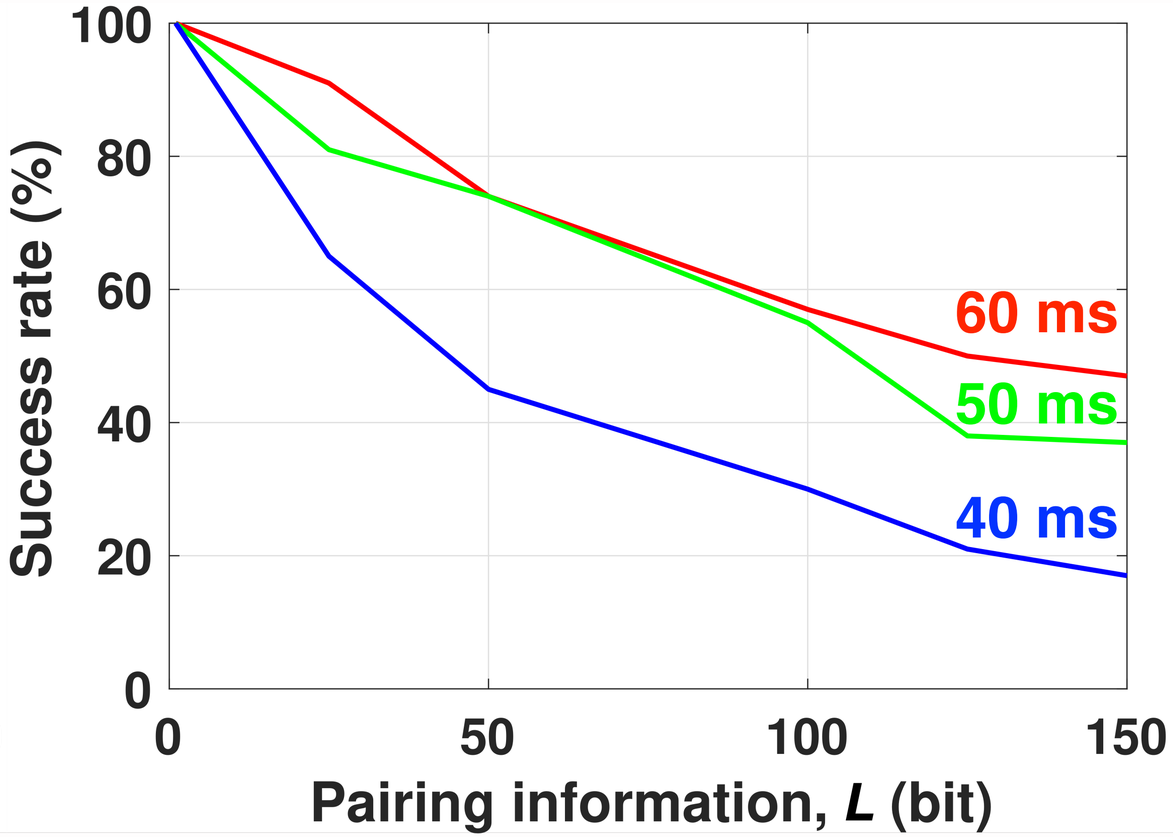


- For each *t*, there exist optimal *k* that maximize expected bps

Pairing Success Rate

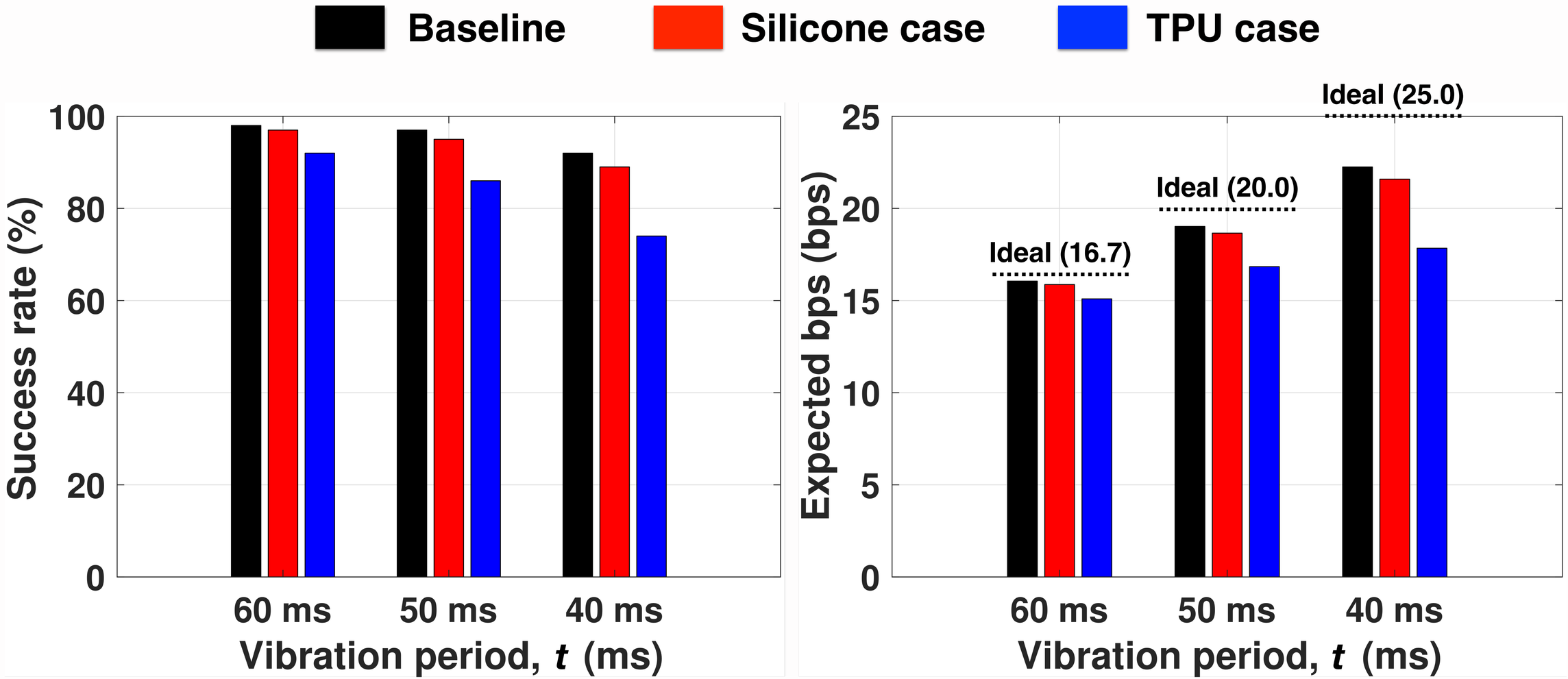


With synchronization

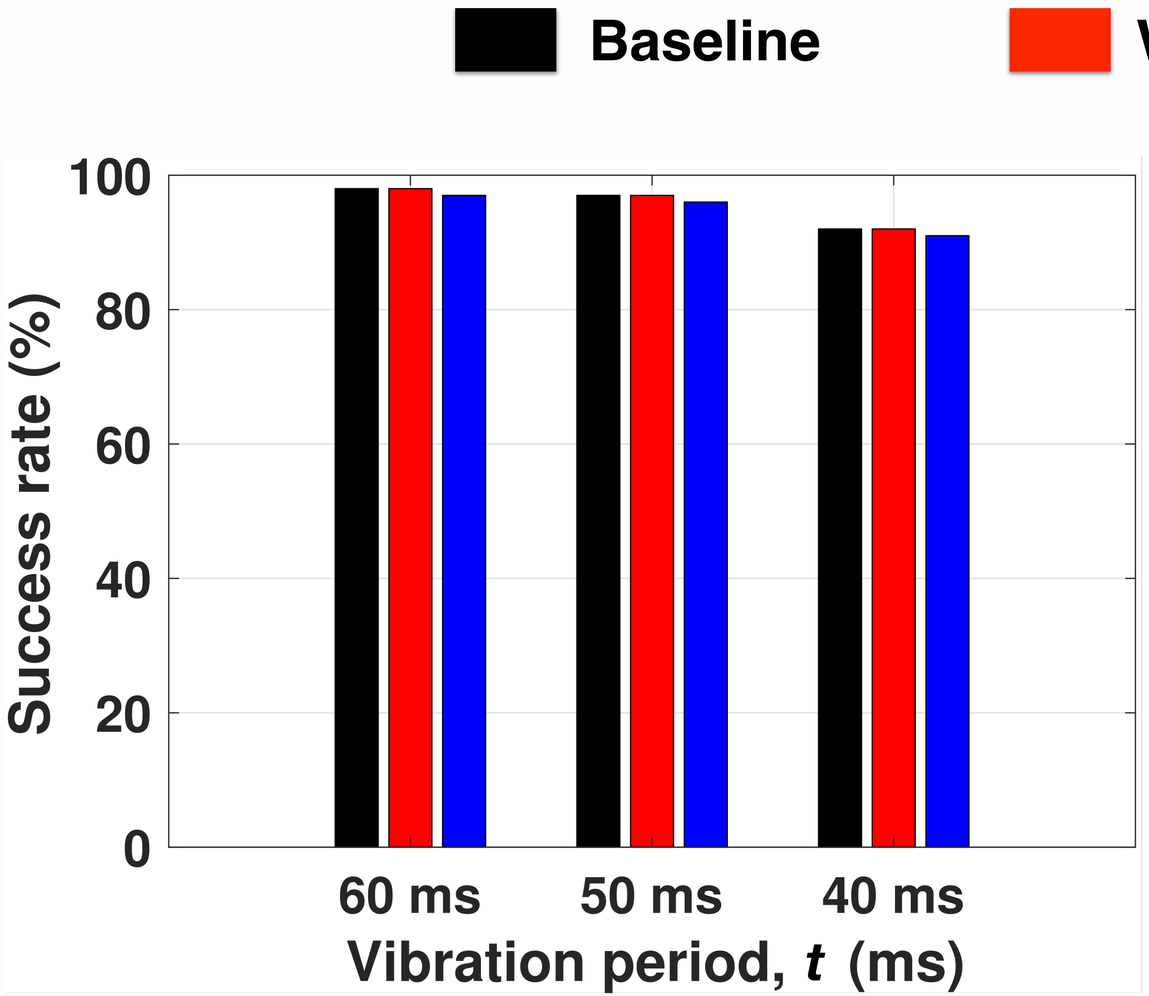


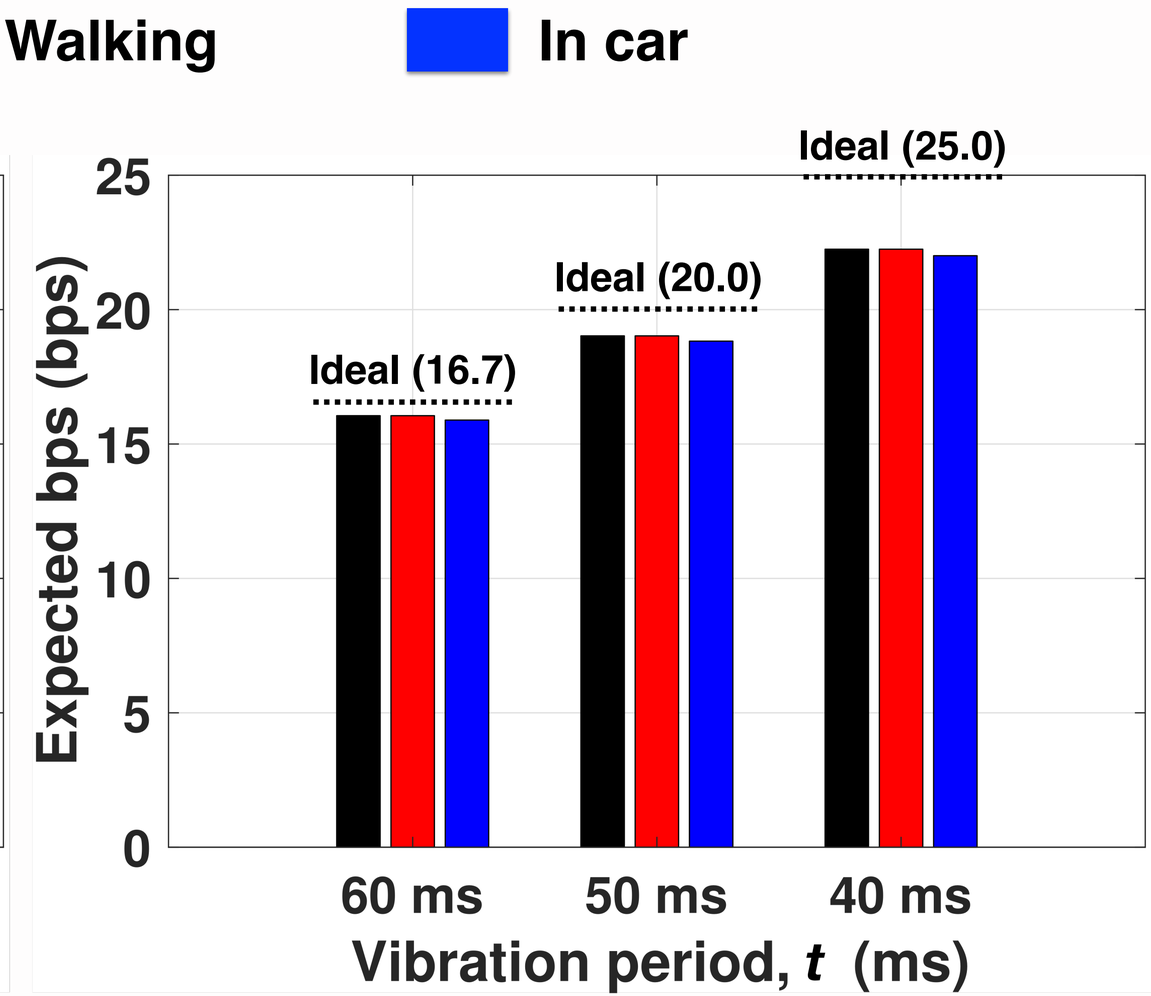
Without synchronization

Transmission Medium



Transmission Environment





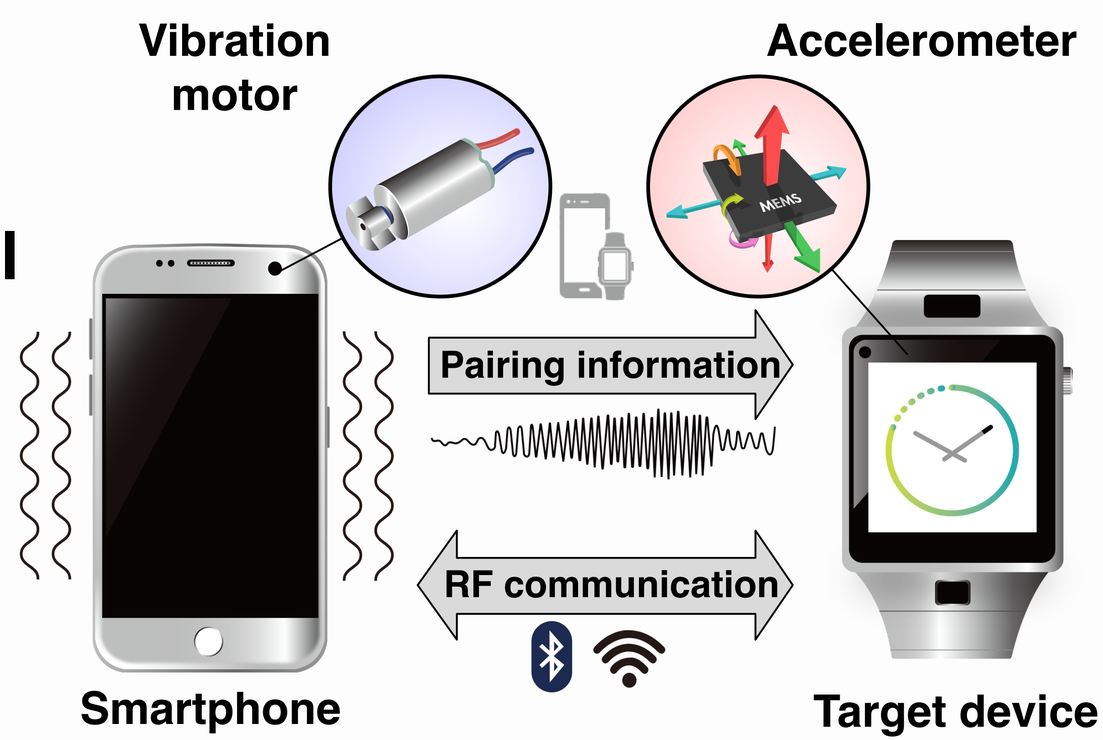
Expected Pairing Time

t	k	Effective bit ratio	Bit error rate	Pairing time
40 ms	30 bits	97.4%	0.95%	6.74 s
50 ms	35 bits	98.2%	0.61%	7.87 s
60 ms	40 bits	98.8%	0.67%	9.34 s

- Average expected pairing time between 6 - 9 s
- Bit-wise error less than 1%

Conclusion

- SyncVibe removes hassle of manual pairing procedure
- Synchronization achieved with minimal insertion of overhead bits
- Proposed scheme is not limited to pairing purposes
- Average pairing time of 6.7 s



Thank You

- Questions?
- Supported by Wisconsin Alumni Research Foundation and NSF

WARF
Wisconsin Alumni Research Foundation

WISCONSIN
UNIVERSITY OF WISCONSIN–MADISON

PURDUE
UNIVERSITY