Obsah

	rotokoly pro řízení přístupu k médiu–deterministické vs. nedeterministické (náhodné čel, hlavní rozdíly, výhody a nevýhody	3
	eterministické protokoly pro přístup k médiu (Polling, Token Ring, Reservation–ased protokoly)–pricnip, efektivita, výhody a nevýhody	4
2.		4
2.	ϵ	5
2.		6
	LOHA protokol–Pure vs. Slotted ALOHA–princip, výhody a nevýhody, pravděpodob	onos
	olize, odvození maximální propustnosti	7
	1 Pure ALOHA	7
3.	2 Slotted ALOHA	9
	SMA (1-persistent, non-persistent, p-persistent, CA, CD)-princip, problémy (Hid-	
	en node, exposed node)	10
4.	1	10
4.	1	11 12
4.	1 1	12
4.		12
4.		14
4.	1	14
5 V	VSN-základní charakteristiky a vlastnosti	15
6 S	měrování pro WSN-proces(discovery,selection,maintanance), základní princip, me-	
tr	riky pro výběr cesty	16
6.		16
	6.1.1 DV–Distance Vector	16
_	6.1.2 LS-Link State	16
	2 Problémy ve WSN	16
6.	•	17
6.	4 Route Selection	17 18
6.	•	19
U.	Noute maintainance	1)
	měrovací protokoly pro WSN a jejich princip-Flooding a jeho optimalizace, Gos-	
	ping, Expanding Ring Search a jeho varianty (Blocking, Two-Side), Low-energy	
	daptive clustering hierarchy 1 Kdy se protokoly aktivují?	19 19
1.	. I Kuy 85 PIUUKUIY AKIIYUJI!	17

		7.1.1 Proaktivní (dopředu si vytvářím routovací tabulky)	19
		7.1.2 Reaktiví (iniciuje zdroj)	19
			19
	7.2		20
	7.3	Flooding	20
	7.4		21
	7.5	Blocking ERS	22
	7.6	Two side ERS	22
	7.7	Fisheye State Routing–FSR	22
	7.8	Gossip	23
		7.8.1 GOSSIP1(p)	23
		7.8.2 $GOSSIP1(p,k)$	23
	7.9	Low Energy Adaptive Clustering Hierarchy–LEACH	23
8	mun kace		24
	8.1	EPC global network (Electronic Product Code)	26
9	Imp	lementace Bluetooth (BR/EDR, LE, AMP)-základní charakteristiky a vlastnosti,	
	výko	onové třídy	27
	9.1	Bluetooth obecně	27
	9.2	Bluetooth Basic Rate/Enhanced Data Rate	28
	9.2 9.3		28 28

1 Protokoly pro řízení přístupu k médiu–deterministické vs. nedeterministické (náhodné)–účel, hlavní rozdíly, výhody a nevýhody

Zde řešíme problém, že několik zařízení chce současně využívat sdílené prostředky (komunikační kanál). Pokud je počet requestů (žádostí o využití sdílených prostředků) vyšší než počet dostupných prostředků, tak to vede k blokování. Pokud více zařízení najednou přistoupí k prostředkům, nastane kolize. Potřebujeme nějaký mechanizmus, který zajistí, že všichni budou vysílat, ale nebude docházet ke kolizím. Proti sobě jdou efficiency vs. overhead. Méně "služebních" (řídicích) informací vede na lepší využití kanálu pro užitečná data (efektivita), ale zase nedosáhnu optimálního rozdělení zdrojů.

Deterministické

- Výsledek rozhodnutí za stejných podmínek je vždy stejný
- Proces výběru můžeme na stejých datech provést kolikrát chceme, ale výsledek bude vždy stejný => tyto modely se chovají "předvídatelně"

V daném scénáři dostane konkrétní uživatel vždy stejné prostředky

- Výsledek závisí na pravidlech daného protokolu
- U těchto protokolů je nutné nějaké řízení, což zvyšuje jejich složitost
- Navíc vyžadují dost přesnou časovou synchronizaci
 Zařízení teda ví, že má vysílat 10 ms v časovém okně, které nastane za X ms, pokud mu ale běží blbě čas, tak začne vysílat v nesprávný čas a nastane kolize
- Výhoda je, že zařízení má zaručený komunikační slot
- Nedeterministické (náhodné)
 - Výsledek rozhodnutí je náhodný

Takže když na stejná vstupní data při stejné situaci v síti pošlem algoritmus rozhodování vícekrát, může rozhodnout pokaždé jinak

V daném scénáři může dostat konkrétní uživatel různé prostředky/nemusí je dostat vůbec

- Výsledek je "nepředvídatelný"
- Asi jednodušší
- Když má zřízení smůlu, musí čekat
- Má výhodu tam, kde je nízký provoz a nezáleží mi na tom, jaké bude zpoždění nebo packet loss (takže pro řízení linky v továrně NE)

2 Deterministické protokoly pro přístup k médiu (Polling, Token Ring, Reservation– based protokoly)–pricnip, efektivita, výhody a nevýhody

2.1 Polling

Centralizovaný kontrolér se každého zařízení zeptá, jestli má něco k odeslání nebo ne.

- Zařízení má data pro odeslání ⇒ data se pošlou
- Zařízení nemá nic k odeslání ⇒ data se neposílají a kontrolér vyzívá další zařízení v pořadí

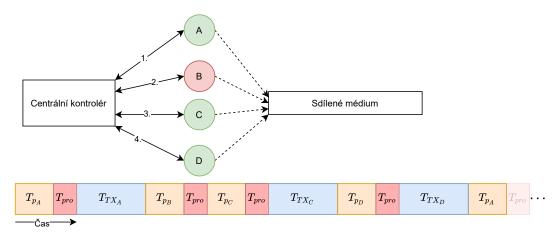
Takže obecně jsou prostředky komunikačního kanálu sdíleny na užitečná data a na řídicí data (výzva–odpověď). Efektivita potom vyplývá z poměru užitečných dat ku řídicím, respektive poměru času vysílání ku času režie a zpracování.

- T_p -polling time
 - čas, který je potřeba k tomu, aby se kontrolér zeptal zařízení, jestli chce, nebo nechce vysílat
- T_{pro} -propagation and processing time
 - čas, který je potřeba k doručení a zpracování odpovědi zařízení
 - tento čas bývá velmi malý a na výslednou efektivitu nemá velký vliv (můžeme zanedbat)
- T_{TX} -transmission time
 - čas, po který dané zařízení vysílá užitečná data

Výsledná efektivita je

$$Efficiency_{polling} = \frac{T_{TX}}{T_p + T_{pro} + T_{TX}}$$
 (1)

- Výhody
 - Dobře využívá kanál (pokud zařízení vysílat neche, tak se nečeká a jede se dál)
 - Nelze médium "přesaturovat"-kontrolér prostě využije maximum a protože se zařízení nejdřív musí zeptat, tak ostatní nevysílají dokud jim to nebude umožňeno
 - * U metod, kde mohou nastat kolize, tak se může stát, že při extrémním počtu zařízení na kanále nastávají pořád kolize a nikdo nic nepřenese
 - Přístup k médiu je předvídatelný-obecná vlastnost deterministických protokolů
- Nevýhody



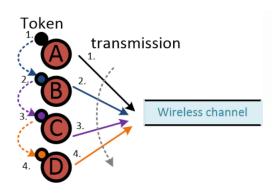
Obrázek 1: Polling

- Ztrácíme čas dotazováním každého zařízení
- Pokud některá zařízení vysíla nechtějí a jedno by chtělo pořád, tak zbytečně plácám drahocenný čas na dotazováné zařízení, které vysílat nechtějí, tento čas by se dal využít k vysílání dat azařízení, která vysílat chtějí
- Řídí to jeden centrální prvek

2.2 Token Ring

Zde je řízení distribuované (nemám žádný centrální kontrolér). Jednotlivá zařízení si předávají **token**, který je opravňuje k vysílání. Takže je to vlastně něco jako Polling, akorát zařízení nekomunikují s centrálním kontrolérem, ale mezi sebou. Zařízení vysílá, když má token, pak ho předá dál (samozřejmě musí vědět komu token předat). Pořadí,ve kterém si token zařízení předávají se nazývá "token ring".

Výsledná efektivita je podobná jako u Pollingu.



Obrázek 2: Token ring

Výhody

- Bez kolizí
- Nemusím mít centrální kontrolér
- Stejně jako u pollingu nemůžu kanál zatížit takovým způsobem, že bude nepoužitelný, prostš dosáhne maxima a hotovo
- Poskytuje velmi vysokou propustnost i pro velmi hustý síťový provoz (je bezkolizní)
- Lze nastavovat priority

Nevýhody

- Jedno zařízení vypadne ⇒ musím nějak potvrdit, že token dorazil, pokud ne, posílám token dalšímu v pořadí
- Složitější řešení možných chyb ⇒ vetší overhead
- Při přidání nového zařízení to musí říct ostatním

2.3 Reservation-based

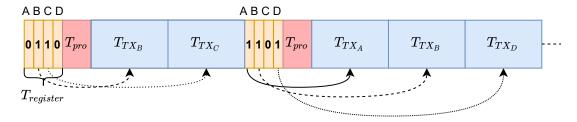
Tady je dobrý podotknout, že tohle není konkrétní protokol, ale jenom mechanismus. Protokolů, kteé tento mechanizmus využívají je hodně (i když fungují všechnz dost podobně). Zde je komunikace rozdělena na dvě části: **Reservation part** a **Communicatio part**.

- Reservation part
 - Rezervační část je veliká úměrně k počtu zařízení
 - V této části zařízení řeknou jestli budou/nebudou vysílat
 - Všechna zařízení tuto rezervační část musí slyšet
 - Pokud budou zařízení vysílat různou dobu, musíme do rezervační části přidat informaci o tom, jak dlouho budou vysílat
- Communication part
 - Zařízení, co chtějí vysílat vysílají v daném pořadí a danou dobu
 - Zařízení slyšely rezervaci ostatních ⇒ ví kdy mají vysílat aby nedošlo ke kolizím

Výsledná efektivita je následující.

$$Efficiency_{reservation} = \frac{T_{TX}}{T_{register} + T_{pro} + T_{TX}}$$
 (2)

- Výhody
 - Jakmile proběhne rezervace a zařízení získá slot, už ho má jistej
 - U některých konkrétních algoritmů se píše, že umožňuje real time QoS
- Nevýhody
 - Zařízení musí být časově přesně synchronizována

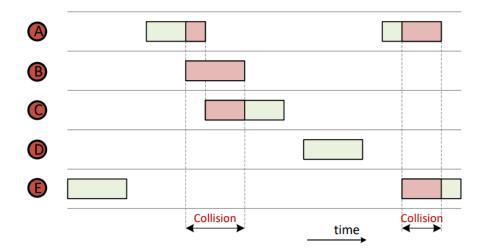


Obrázek 3: Reservation based protokol

3 ALOHA protokol-Pure vs. Slotted ALOHA-princip, výhody a nevýhody, pravděpodobnost kolize, odvození maximální propustnosti

3.1 Pure ALOHA

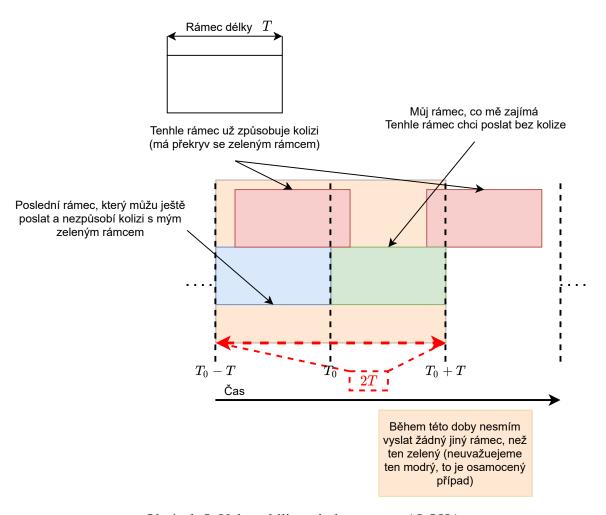
Pure ALOHA je totální chaos – zařízení prostě pošle data, když nějaký má a nikoho se na nic neptá. Jakmile nějaké jiné zařízení udělá to samé, vzniká kolize. Tento režim se snad ani nedá nazvat protokolem, prostě všichni spamujou ⇒ super jednoduchý, ale málo efektivní.



Obrázek 4: Pure ALOHA

Odvození maximální propustnosti má určité předpoklady:

- Všechna zařízení posílají rámce stejné délky T
- Pravděpodobnost generace k rámců během času T je dána Poissonovým rozdělením $P[k] = \frac{\lambda^k e^{-\lambda}}{k!}$
 - λ-očekávaný počet vygenerovaných rámců
- Aby spolu rámce nekolidovali, nesmí se objevit další rámec v intervalu 2T, který začíná 1T před mým rámcem–vim, že to neumim vysvětlit, takže tady je obrázek č. 5



Obrázek 5: Vulnerability window u pure ALOHA

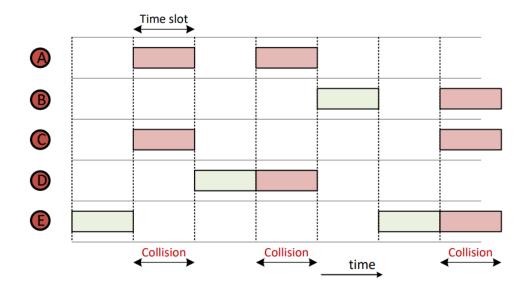
- $S = \lambda P_0$
 - S značíme jako propustnost
 - $P_{0_{1T}}$ je pravděpodobnost, že se vygeneruje právě 0 rámců za čas 1T $P_0=\frac{e^0e^{-\lambda}}{0!}=e^{-\lambda}$
- Já chci interval 2T, kde se nebudou generovat žádné rámce $\Rightarrow P_{0_{2T}} = e^{-2\lambda}$
 - Tedy počet očekávaných rámců ze z λ zvedne na 2λ
- Propustnost: $S = \lambda e^{-2\lambda}$
 - Maximální propustnost zjistíme derivací propustnosti
- Maximální propustnost: $S_{max}=\frac{dS}{d\lambda}=e^{-2\lambda}+\lambda(-2)e^{-2\lambda}\Rightarrow (e^{-2\lambda})(1-2\lambda)=0$

–
$$\lambda = \frac{1}{2} \Rightarrow$$
 maximální propustnost

•
$$S_{max} = \frac{1}{2}e^{-2\frac{1}{2}} = 0,184$$

3.2 Slotted ALOHA

Slotted ALOHA už je menší chaos, tudíž má i lepší propustnost. Vysílání je synchronizováno a vždy začíná na začátku timeslotu. Takže už se nestane, že by se rámce překrývaly jen částečně, teď se překrývají úplně nebo vůbec. Toto vyžaduje přesnou časovou synchronizaci mezi jednotlivými zařízeními. Odvození maximální propustnosti má určité předpoklady (stejné jako u pure ALOHA).



Obrázek 6: Slotted ALOHA

- Všechna zařízení posílají rámce stejné délky T
- Pravděpodobnost generace k rámců během času T je dána Poissonovým rozdělením $P[k] = \frac{\lambda^k e^{-\lambda}}{k!}$
 - $-\lambda$ -očekávaný počet vygenerovaných rámců
- Aby spolu rámce nekolidovali, musí okolo sebe mít místo alespo
ň ${\cal T}$ na každou stranu
- $S = \lambda P_0$
 - -S značíme jako propustnost
 - $P_{0_{1T}}$ je pravděpodobnost, že se vygeneruje právě 0 rámců za čas 1T $P_0=\frac{e^0e^{-\lambda}}{0!}=e^{-\lambda}$
 - Tady je pro interval 1T už pravděpodobnost vyhovující, protože tím, že jsou časy vysílání synchronizované se mi zkrátilo kritické okénku z2T na 1T

- Propustnost: $S = \lambda e^{-\lambda}$
 - Maximální propustnost zjistíme derivací propustnosti
- Maximální propustnost: $S_{max}=\frac{dS}{d\lambda}=e^{-\lambda}-\lambda e^{-\lambda}\Rightarrow (e^{-\lambda})(1-\lambda)=0$
 - $\lambda = 1 \Rightarrow$ maximální propustnost
- $S_{max} = e^{-1} = 0,368$

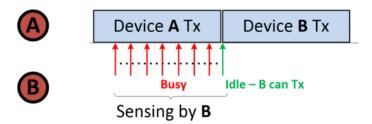
4 CSMA (1-persistent, non-persistent, p-persistent, CA, CD)-princip, problémy (Hidden node, exposed node)

CSMA = Carrier Sensing Multiple Access

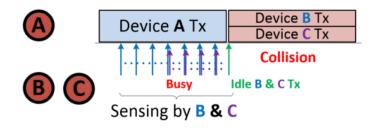
Obecně čeká, až bude médium volné (Carrier Sensing) a až pak začne vysílat. Tím se omezí kolize, ale bezkolizní to úplně nebude.

4.1 1-persistent

Pokud je kanál zaneprázdněný, tak zařízení čeká. Jakmile se kanál uvolní, začíná zařízení vysílat. To ale znamená, že jakmile se uvolní kanál, mohou všechna čekající zařízení začít vysílat najednou \Rightarrow kolize (zařízení si myslí, že je kanál volný, ale jiné zařízení už začlo vysílat, jenom se to k tomuto zařízení ještě nedostalo). V případě, že dojde ke kolizi, tak obě zařízení čekají náhodně zvolený čas a pak zase "měří". Tento režim je vhodný pouze pokud máme málo zařízení, nebo zařízení nekomunikují často.



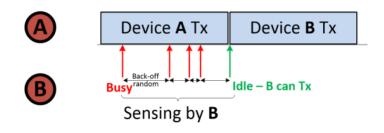
Obrázek 7: Případ bez kolize



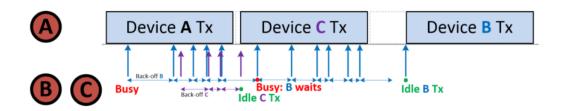
Obrázek 8: Případ s kolizí

4.2 Non-persistent

Vylepšení oproti 1-persistent. Zařízení neposlouchá kontinuálně, ale pouze po uplynutí náhodné doby. Když zařízení zjistí, že je kanál obsazený, tak si zvolí náhodný časovač a po tomto intervalu kanál změří znovu, je-li volný. Pokud po této době je kanál volný, začne vysílat. Pokud není, tak znovu zvolí náhodný časovač a zase po tomto intervalu měří kanál. To způsobí to, že čekající zařízení měří kanál v různých časech \Rightarrow snižuje se pravděpodobnost kolizí. Nevýhoda je v tom, že kanál může být volný, i když by někdo mohl vysílat, ale protože měl smůlu a zvolil si časovač moc dlouhý, tak kanál změří až za chvíli a v této mezeře je kanál nevyužitý.



Obrázek 9: Ukázka náhodných intervalů

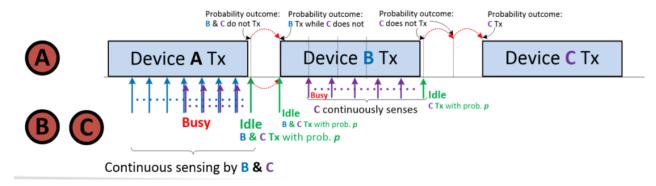


Obrázek 10: Ukázka zbytečných prodlev kvůli časovačům (nekontinuální měření)

4.3 p-persistent

Měří se kanál vždy na začátku time slotu a pokud je volný, začne se vysílat s pravděpodobností p, respektive i když je kanál volný, vykašli se na vysílání s pravděpodobností 1-p. Pokud se změří, že je kanál obsazený, nic se neděje a měří se dál pořád stejně (periodicky na začátku time slotu). Čekající zařízení počkají až do okamžiku, kdy je kanál volný a potom každé zařízení začne vysílat s pravděpodobností $p \Rightarrow$ takže aby začaly vysílat všech n zařízení, tak bych musel vyhrát sportku s pravděpodobností p^n , kde p je samozřejmě menší než 1 (násobení čísel menších než 1 jde do háje rychle).

Má to nižší pravděpodobnost kolize než 1–persistent, navíc pravděpodobnost kolize lze ovládat parametrem p. Vetší p znamená, že nebudu tak dlouho čekat, než se zařízení "chytí" (vyhraje maty s pravděpodobností p), ale zase je vyšší pravděpodobnost kolize (víc zařízení vyhraje, pokud zvýším p). Menší p znamená, že se sníží kolize, ale zase se zvýší doba, než někdo začne vysílat, protože s malou pravděpodobností p, budou "vyhrávat" méně často.



Obrázek 11: P-persistent

Na obrázku je vidět, že potom,co odvysílá A se jeden slot nevysílá, portože to nepadlo ani B ani C. Další slot už B padne a C tedy znovu čeká. Po skounčení vysílání má C smůlu a vysíá až za 2 sloty.

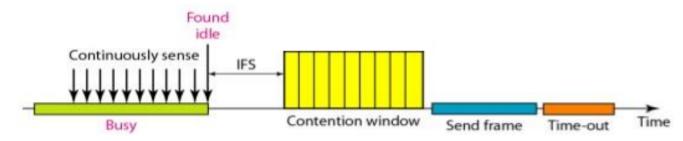
4.4 CD-Collision detection

Je to založené na Carrier Sensing (měření kanálu). Pokud je detekována kolize, všechna zařízení přestanou vysílat (při kolizním stavu je to zbytečné). Problém je u bezdrátových sítí, kde většina zařízení pracuje v half–duplex režimu, tedy neumí poslouchat a vysílat zároveň. Kdybych měl 2 antény, tak bych si tou vysílací hezky rušil tu přijímací ⇒ problém není jednoduchý na vyřešení.

4.5 CA–Collision Avoidance

Sense before transmit–pokud je kanál obsazen, nevysílej (tohle používá WiFi). Pokud kanál obsazen není, tak počkej doub **IFS**. IFS–Inter Frame Space slouží ke kompenzaci zpoždění signálu způsobené konečnou rychlostí šíření. Tedy toto zabraňuje situaci, kdy se informace o tom, že některé zařízení již vysílá nedostane k zařízení na vzdáleném konci sítě, tudíž ono vzdálené zařízení

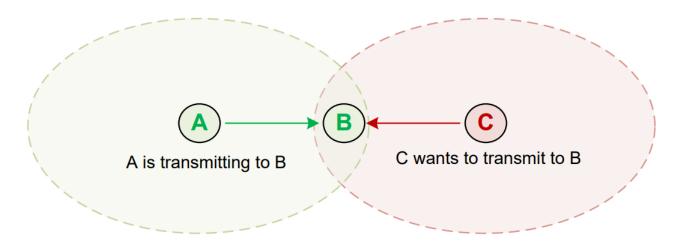
si myslí, že je kanál volný a vesele si začne vysílat, což vede ke kolizi. Pokud po uplynutí doby IFS je kanál stále volný, počkej na **Contention Window**, což je náhodná doba (počet slotů) sloužící ke snížení pravděpodobnosti kolizí. Pokud po vyslání rámce nedostaneme potvrzení (ACK) do timeoutu, pošli rámec znovu.



Obrázek 12: Collision Avoidance

4.6 Hidden node problem

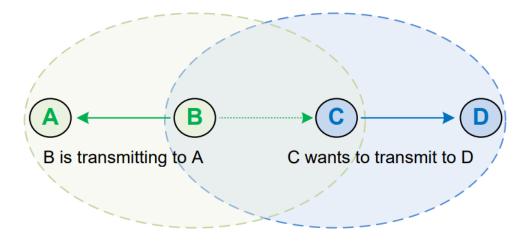
Zvyšuje počet kolizí pro všechny metody CSMA. Mějme 2 zařízení A a C, v jejichž oblasti dosahu se protínají v místě zařízení B a zároveň zařízení A a C jsou mimo vzájemný dosah (nevidí na sebe). Takže A neví o existenci C a C neví o existenci A. Dejme tomu, že A komunikuje s B a C chce také komunikovat s B. Protože C je mimo dosah A, tak zřízení C zařízení A "neslyší" a myslí si, že kanál je volný a zače tedy vysílat směrem k B. U B se sejdou signály jak od A tak od C a dojde ke kolizi. Možné řešení by bylo, kdyby B nějak rozšířilo informaci o tom, že A vysílá (ale asi celkově ten problém bude velmi složitý, kdyby to bylo jednoduchý ho vyřešit, tak se to asi neučíme že jo xd).



Obrázek 13: Hidden node

4.7 Exposed node problem

Snižuje využití kanálu pro všechny metody CSMA. Mějme 4 zařízení A,B,C a D rozmístěny podle následujícího borázku č.14. B vysílá k A a C chce vysílat k D. Protože C je v dosahu B, tak slyší komunikace dvojice BA. Pokud by C začlo vysílat k D, tak A neovlivní a D zase neuslyší B (takže win–win), ale to se nestane, protože C slyší komunikaci AB a tak si myslí, že nemůže vysílat,že je komunikační kanál "obsazen" (což ve skutečnosti je, ale zde by nevadilo, kdyby vysílalo B k A a C k D zároveň). Tímto zbytečne plýtvám prostředky (nevysílám, i když bych mohl)



Obrázek 14: Exposed node

5 WSN-základní charakteristiky a vlastnosti

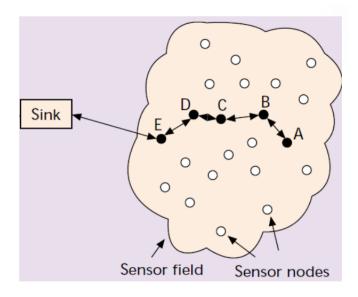
WSN–Wireless Sensor Networks. Tyto sítě jsou charakteristické tím, že se skládají z uzlů (typicky senzorů) distribuovaných na nějaké ploše, kde tyto uzly si navzájem vyměňují informace, nebo informace posílají někam dál. Číselně se pohybujeme od desítek, stovek až do tisíců zařízení v síti. Lze přenášet nějaká jednoduchá data (jako třeba teplotu) nebo třeba i obraz, ale ten se nepřenáší kontinuálně (na to jsou jiné druhy sítí). Uspořádání může být strukturované (předem víme kde senzory budou) nebo nestrukturované (senzory jsou naházeny random). Je to síť složená z mnoha většinou bateriově napájených malých zařízení, takže by měla být dobře škálovatelná a musí si umět poradit s výpadkem jednotlivých zařízení.

Každý síťový uzel (senzor) může být charakterizován jako

- Zdroj
 - Generuje data k odeslání
- Cílové zařízení
 - Data zde končí, přijímá data od Zdrojového uzlu
- Relay
 - Přeposílá data dál (směrování od Zdroje k Cílovému zařízení)

Každý uzel by měl být:

- Levný
- Nízkopříkonový
 - Měl by tedy dlouho vydržet na baterii
 - To omezuje výpočetní schopnosti zařízení a taky komunikační dosah



Obrázek 15: WSN

6 Směrování pro WSN-proces(discovery, selection, maintanance), základní princip, metriky pro výběr cesty

6.1 Druhy směrování

6.1.1 DV-Distance Vector

Zde se v routovací tabulce udržuje pro daný cíl Next Hop (směr vektoru) a Hop Count (velikost vektoru). Vybírá se cesta s nejmenším počtem hopů, což je sice velmi jednoduché, ale nemusí to vždy sledovat nejvýhodnější cestu (nejkratší cesta může být zrovna přetížená, má špatnou kvalitu spojení atd.).

6.1.2 LS-Link State

Zde se rozhoduje na základě tzv. metriky. Každý spoj se ohodnotí nějakou metrikou a pak se vybírá spoj s nejmenší metrikou. Metrika lze počítat různě. To tedy zohledňuje neenom vzdálenost, ale i aktuální stav sítě. Zde je nevýhoda v tom, že je zde velký overhead.

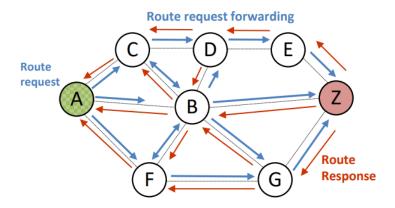
6.2 Problémy ve WSN

Kanál se dynamicky mění, senzory většinou nemají nějaké přizpůsobovací směrové antény, dochází k tomu, že cesta z A do B má jiné vlastnosti kanálu než z B do A (odrazy, zakrytí, rušení...). Máme omezené prostředky pro signalling, což se váže i na omezení spotřeby–chceme aby zařízení taky něco vydrželo. V základní formě DV ani LS nejsou pro WSN vhodné a jsou nutné úpravy.

6.3 Discovery

Nalezení možných cest mezi zdrojovými a cílovými uzly sítě. Zde může být výběrovým parametrem třeba maximální počet skoků, zpoždění atd.

- Posílá se route request paket okolním sousedům
- Sousední uzly tento route request přeposílají dále svým sousedům
 - Takhle pořád dokola, až se to dostane k cíli
 - Kdo pak bude směrovat a jak záleží na konkrétním protokolu
 - Při mapování cesty se rovnou mhohou ukládat informace o parametrech cesty
- Cíl zjistí, že ho někdo hledá a pošle **route response** po té samé cestě, po jaké se k němu dostal request



Obrázek 16: Route Discovery

6.4 Route Selection

Nalezené cesty se ohodnotí a vybere se ta nejvhodnější pro naši aplikaci. Cesta r se vybírá na základě metriky m.

- Minimální metrika: $r = minm_i$
 - Maximální úspora energie
- Maximální metrika: $r = maxm_i$
 - Maximální průchodnost
- Můžu klást ještě další omezení jako třeba:
 - maximalizuj průchodnost, ale spotřeba nepřesáhne danou hranici

- minimalizuj spotřebu energie, ale neklesneš pod danou propustnost
- Typicky se vybí rá trasa, kde minimalizujeme spotřebovanou energie, ale zpoždění nepřesáhne určitou mez.

Výsledná metrika lze spočítat celkem jak si to vymyslíme, určitě to ale bude nějaká kombinace následujících parametrů.

6.4.1 Metriky

- Spotřeba energie
 - Vyjadřuje se ve spotřebě v Joulech na jeden bit
- Network life-time
 - Čas, který uplyne než odejde první uzel. Chováme se tak, že chceme aby ideálně všechnz uzly odešly naráz ⇒ balancujeme vytížení jednotlivých uzlů.
- Delay
 - Čas mezi vyslání informace zdrojem a příjmem informace cílem, mezilehlá zařízení zavádějí zpoždění
- Packet delivery ratio (spolehlivost)
 - Poměr mezi oručenými pakety do cíle a všemi poslanými pakety zdrojem v nějakém časovém okně
 - * Důležitý pro komunikaci mezi autama, paket MUSÍ dorazit
- Route stability
 - Monitorování metriky dané cesty-metrika se mění v čase (kvalita kanálu, stav baterie uzlu atd.)
 - * Čím stabilnější cestu máme, tím méně často si o ní musíme zjišťovat informace
- Manbagement cost (overhead)
 - Režijní data určená pro řízení (signalling, route discovery...)
- Throughput (propustnost), data rate...
 - Mnoství přenesené informace za jednotku času

6.5 Route maintanance

Udržujeme informace o dané cestě aktuální. Jsou to informace o mibilitě, rušení, nových uzlech, stavu baterie atd.

- Proměnná dostupnost cest
 - Dostupnost ověřujeme tím, že vyžadujeme potvrzování paketů (ACK)
- Proměnná kvalita cesty
 - Můžeme si spočítat útlum (víme vysílací výkon)
 - Kvalitu kanálu nahlašujeme sousedům

Tyhle informace nám pomáhají dynamicky vybírat vždy tu nejvhodnější cestu. Vždy jde proti sobě aktuálnost informací o cestách a overhead (režijní data). Musíme tedy najít balanc mezi frekvencí zjišť ování informací o síti a přenášením užitečných dat.

7 Směrovací protokoly pro WSN a jejich princip–Flooding a jeho optimalizace, Gossiping, Expanding Ring Search a jeho varianty (Blocking, Two–Side), Low–energy adaptive clustering hierarchy

7.1 Kdy se protokoly aktivují?

7.1.1 Proaktivní (dopředu si vytvářím routovací tabulky)

Protokoly periodicky aktualizují směrovací tabulku, i když žádná komunikace neprobíhá (není vyžadována). Dalším spouštěčem aktualizace routovací tabulky může být změna v topologii sítě, změna kvality kanálu, úrovně baterií atd. Je tam jsitý overhead, protože se updatuje periodicky \Rightarrow vyšší spotřeba energie. Takže tohle použiju pro sítě, kde potřebuju **malý delay**.

7.1.2 Reaktiví (iniciuje zdroj)

Cesty se hledají pouze pokud je potřeba. Tím zvniká zpoždění a overhead, protože se cesta naëjdřív musí nají t a pak vybrat. Zařízení tedy dopředu neví jakou cestu bude mít. Není tam overhead ve smyslu toho, že by se aktivovaly protokoly, když nikdo nekomunikuje–když nikdo nekomunikyuje protokol nic nedělá \Rightarrow šetří energii. Takže tohle potřebuju pro sítě, kde je prioritní **low power consumption**.

7.1.3 Hybridní

Kombinace obou předešlých.

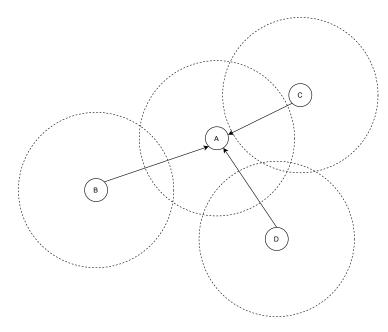
7.2 Topologie sítě

- Flat
 - Všechny uzly jsou si rovny
- · Hierarchical
 - Některá zařízení mají různé funkce
 - "Cluster head"/gateway jsou řídicí body dané oblasti (místní správci sítě)
 - Síť je rozdělena do clusterů
- · Location based
 - Protokoly mohou využít známou polohu zařízení (směrování ≈ geometrie)

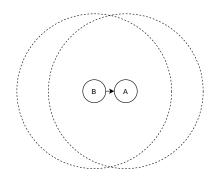
7.3 Flooding

Je to nejjednodušší protokol (no nazývat to protokolem je skoro urážka, HUB není protokol Change My Mind 2022). Zařízení, co přijme zprávu, je automaticky přeposílá všem sousedům. Zde se nevyužívá žádných informací o síti–žádné routovací tabulky, informace o cestách atd. Obrovský overhead, protože zprávy se šíří všude, ale cíl je třeba jenom jeden (ale jako broadcast dobrý xD). Vylepšení floodigu:

- Single message repetition only
 - Nepřeposílej tu samou zprávu dál ⇒ když zdroj vyšle, tak sousedi mu to pošlou zpátky, zdroj potom tuto zprávu již nešíří dál ⇒ zabraňuje se tím zpětnému šíření z "znovuposílání" od zdroje.
- Reception from multiple-directoions
 - Pokud tu samou zprávu dostaneš z více směrů, nepřeposílej to dál
 - * Tady se uvažuje, že sousedi to maj dost dobře pokrytý, takže není třeba přeposílat, viz obrázek 17–zde A nebude přeposílat zprávy od ostatních (pokud jsou stejné).
- Close nodes
 - Pokud víme, že jsou 2 uzly velmi blízko, nemá význam to přeposílat z obou ⇒ vybere se jenom jeden z nich
 - * Tedy neposílej data dál, pokud si je přijal od zdroje, který je velmi blízko tebe samotného (dá se totiž předpokládat, že já i onen zdroj máme podobné pokrytí), viz obrázek 18– A to nebude přeposílat dál.
 - Blízkost se určí ze znalosti polohy uzlů, nebo na základě síly signálu



Obrázek 17: Flooding: Multi reception



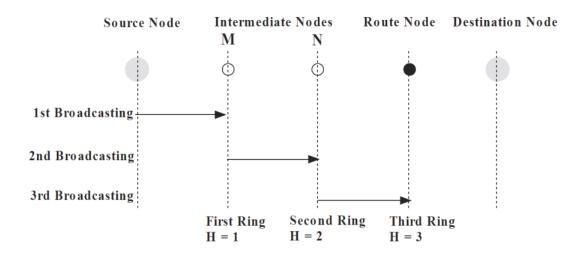
Obrázek 18: Flooding:Close nodes

7.4 Expanding Ring Search-ERS

Využívá se parametru tzv. **TTL**—**T**ime **T**o **L**ive. Tento parametr definuje, kolik skoů může paket udělat před tím, než je zahozen (jakmile je TTL=0, tak se paket zahozuje, toto zabraňuje nekontrolovanému bloudění paketu v síti). Ze začátku se začíná s TTL=1 (tudíž dostane se to pouze k nejbližším sousedům) a pokud paket nedojde do cíle, postupně se TTL navyšuje. Nevýhoda je v tom, že **musíme čekat** na odpověď. Vzhledem k tomu, že čekáme na odpověď, jestli už paket s daným TTL nedorazil, tak musíme čekat s ohledem na zktuální hodnotu TTL (pro malé TTL stačí čekat kratší dobu než pro velké TTL). Funguje dobře v malých sítích, ve velkých sítích už se nevyplatí hledat cestu tímto způsobem–hodně pokusů se zvyšováním TTL = velká spotřeba enerie.

7.5 Blocking ERS

Stejné jako normální ERS, ale zdroj znovu neposílá data s vyšším TTL. Místo toho další zařízení (dál od zdroje) sama přeposílají data a sama tak najdou cíl. Zařízení samo pozná, jestli je cílem, nebo není. Pokud zařízení není cílem, automaticky zprávu přeposílá dál. Pokud zařízení je cílem, automaticky odpovídá zdroji. Jakmile je cíl nalezen, tak zařízení posílají zprávu zdroji, že se cíl našel a zdroj vysílá zprávu STOP všem dalším zařízením, aby již zprávy nepřeposílala (nehledala cíl). Výhoda je v tom, že ušetříme počet vysílání, nevýhoda je v tom, že mám velké zpožděníotázka je jak dlouho bychom měli čekat (čekací doba se definuje jako dvojnásobek TTL, 1 TTL je zde bráno jako doba přenosu mezi sousedními zařízeními).



Obrázek 19: Blocking ERS

7.6 Two side ERS

ERS, které jde "z obou stran". Vyhledává tedy jak zdroj, tak cíl najednou "proti sobě". Zde je důležité zmínit předpoklad, že zdroj a cíl o sobě vědí. Výsledný počet uzlů ke směrování potom vyjde nižší než u "klasického ERS".

7.7 Fisheye State Routing–FSR

Je to založené na tom, že routovací informace o blízkých sousedech si aktualizuju často. Čím vzdálenější uzel, tím méně časo si aktualizuji jeho routovací informace, protože je náročné tyto aktualizace dělat často (energeticky, časově). Zdroj vyšle data směrem na základě své routovací tabulky. Na každém hopu se použije routovací taqblka daného hopu, protože ten aktuální hop má nejaktuálnější informace o routování kolem. Každý uzel tedy upraví cestu, kudy se mají data vydat.

7.8 Gossip

Je to takový vylepšený flooding. Je to založené na tom, že někdo zprávu bude šířit a někdo ne.

7.8.1 **GOSSIP1(p)**

- Zdroj vyšle data s pravděpodobností p=1
- Uzel, co zprávu přijme ji přeposílá s pravděpodobností p (pokud p==1, pak se jedná o flooding)
 - Uzel zprávu nepřepošle s pravděpodobností 1-p
- Pokud uzel t samou zprávu obdrží vícekrát, automaticky ji zahazuje
- Problém může nastat, když je mezi zdrojem a cílem málo uzlů ⇒ může se stát, že se zpráva nedoručí

7.8.2 GOSSIP1(p,k)

Prvních k skosů je pravděpodobnost přeposlání vždy p=1, pak už normální gossip $(p\neq 1)$. Otázka je jak nastavit konstanty p,k.

7.9 Low Energy Adaptive Clustering Hierarchy-LEACH

Zlepšuje network life–time, velká životnost sítě. Je to Cluster–based protokol. To znamená, že uzly posílají zprávy zařízení, kterému se říká "cluster head". Cluster head potom komunikaci uvnitř svého clusteru řídí pomocí time division multiplexingu a komunikace mezi clustery se dělá přes mezilehlé clustery. Role clusteru postupně rotuje, takže se hezky rozloží energetické zatížení zařízení. Jedná se o distribuovaný a spolehlivý algoritmus. Proces selekce cluster head je následující:

- Definujeme si, kolik má být cluster headů \Rightarrow definujeme si číslo P (0-1), které značí podíl cluster headů
 - V každém kole cluster heady vybíráme náhodně
- Každý uzel vyjádří svou "ochotu" stát se cluster headem, naví si vygeneruje náhodné číslo (0–1)

$$T(n) = \begin{cases} \frac{P}{1 - P \cdot (r \mod(\frac{1}{P}))} & pokud \quad n \in G \\ 0 & jinak \end{cases}$$

- * G-uzly, které nebyly cluster head posledních $\frac{1}{P}$ kol.
 - · neboli pokud bylo zařízení Cluster Headem v posledních $\frac{1}{P}$ kolech, pak se cluster headem stát nemůže
- * r-číslo kola (startuje se s 0)

- * pokud $rand < T(n) \Rightarrow$ uzel n se stává cluster headem
- Cluster Head rozdistribuuje ohlášení, že on je nyní novým cluster headem
- Ostatní uzly jsou přiděleny ke cluster headům na základě síly signálu (nejsilnější signál=pravděpodobně nejbližší cluster head–k tomu se připojím)

8 Technologie radiofrekvenční identifikace (RFID)-princip činnosti, transpondéry, komunikační řetězec, architektura včetně serverového zázemí (EPC Global), ONS, aplikace

Cokoliv, co má anténu je v podstatě RFID, přičemž zařízení jsou tagy a čtečky. Data z tagu jsou čtena bezdrátově čtečkou (RF), tedy tag se čtečce nějak identifikuje (ID). RFID čip může být ve variantě pro čtení i pro zápis. RFID je provozováno na následujících frekvencích:

- LF 125–134 kHz
 - Imobilizéry v autě (keyless entry)
 - Animal-ID
 - Access control-čipová karta na vstup (ISIC)
 - Vzdálenost čtení je v řádu centrimetrů, rychlost čtení je malá
 - Jsou odoloné proti "RFID opaque" materiálům: voda, kovy umí se dostat zkrz tyto materiály
 - * Proto se využívá pro Animal–ID nebo pro ovoce
- HF 13,56 MHz
 - Dosah centimetry až metr
 - Tady funguje třeba NFC
 - Relativně vysoká čtecí rychlost
 - Logistika, Access Control, zoží v obchodě, zavazadla, pošta, knihovny
- UHF 868 MHz v Evropě, jejvětší snímací vzdálenost (1–10 metrů)
 - Některé UHF využívají i 2,45 a 5,6 GHz (lokalizace osob, car location, transport, train location)
 - Trackování palet, kontejnerů

Dělení na pasivní a aktivní tag.

- Pasivní tag
 - Tento tag nemá baterii a je čistě napájen z energie pole vysílaného čtečkou, malý vysílací výkon

Aktivní tag

- Má vlastní zdroj energie (baterii), vysílací výkon tak může být větší a můžeme dosáhnout na větší vzdálenosti, tag je pak logicky větší adražší
- Dají se dělit na transpondéry (komunikují pouze v blízkosti čtečky) a becony (komunikují-broadcastují samy o sobě)
- Semi–aktivní tag
 - V zákaldu pasivní tag s malou baterkou
 - Baterka mu pomáhá jenom někdy-není to hlavní zdroj (je to "battery-assisted")

Co je to 1-bit EAS?

- Je to Electronic Article Surveilance
- Tímto lze pouze poznat, jestli tam tag je, nebo není (používá se v obchodech, aby lidi nekradli)
- Když se Tag nachází v poli vysílače, tak začne spotřebovávat energii (tag tvoří rezonanční obvod). Toho si všimne vysílač/přijímač a tím detekuje přítomnost tagu.

Co je to 1–Microwave?

- Je to zase systém ,který dokáže poznat, jestli tam tag je nebo není.
- Tímto lze pouze poznat, jestli tam tag je, nebo není (používá se v obchodech, aby lidi nekradli)
- Tag funguje jako násobička frekvence, tudíž přijímač pozná změnu frekvence (toho že tam je najednou nějaká harmonická, která tam dřív nebyla)

Způsoby přenosu informace jsou následující:

- Indukční vazba
 - pokužívá se pro frekvence 125 kHz a 13,56 MHz
 - Tag se napájí z pole a energii si udržuje v kondenzátoru. Informace se přenáší tak, že se přijímací cívka buď vyzkratuje nebo ne ⇒ na vysílači poznám tyto zátěžové impulsy.
- Elekctromagnetic backscatter

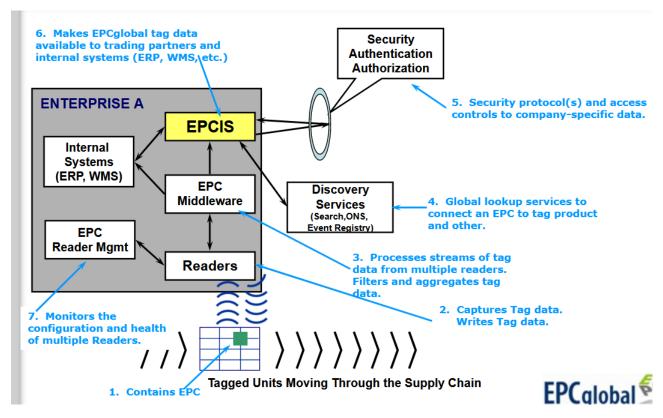
- pouze u UHF
- Tag je schopen se impedančně přizpůsobovat–mění tak koeficient odrazu a tím komunikuje s vysílačem
- SAW-Surface Acoustic Wave
 - No tak to xD, nějak to nechápu, nemělo by to být wireless? xd

8.1 EPC global network (Electronic Product Code)

EPC global je oraganizace, která vymýšlí standardy pro RFID systémy. EPCglobal Network je síť, kde se sdílí data o produktech mezi obchodníky. Každé zařízení má unikátní identifikátor **EPC**–Electronic **P**roduct **C**ode. EPC se skládá ze 4 základních částí:

- Object naming service-ONS
 - Je to služba, která zajišťuje vyhledání ("discovery") informací o objektu na základě jeho EPC
 - S daným EPC je spojena nějaká URL nebo IP adresa, která je dohledatelná
 - * je to něco jako DNS pro EPC
 - Pomocí ONS jsou lokalizovány všechnz databáze, ve kterých se ono EPC nachází
 - Čtečka naskenuje kód → EPC je posálo middlewaru, který se dotáže ONS → ONS ukáže na server, kde jsou uložena data o daném produktu (middlewarwe pak tyto informace stáhne a pnasměrovat informace, kam je potřeba)
- Discovery services
 - Jsou to nástroje jak vyhledávat v síti, je to něco jako search engine
- Information services
 - Je to služba, která umožňuje sdílet data mezi různorodými aplikacemi a napříč podniky
 - prostřednictvím tohoto systému lze zjistit, jaký produkt se v danou dobu vyskytoval na daném místě a z jakého důvodu
- Security services
 - Nástroje, které poskytují zabezpečený přístup do EPCglobal Network

Aplikace jsou halvně v logistice, v obchodech, pasy, automobilový průmysl...-do budoucna se uvažuje, že téměř vše bude otagováno.



Obrázek 20: EPC global

9 Implementace Bluetooth (BR/EDR, LE, AMP)–základní charakteristiky a vlastnosti, výkonové třídy

9.1 Bluetooth obecně

Pracuje v pásmu 2,4 GHz, přičemž rychlost přenosu se pohybuje kolem 1–3 Mbps. Podporuje point–to–point i point–to–multipoint spojení. Zvládne až 8 připojených zařízení najednou. Využívá tzv. frequency hoppingu (každou chvíli mění vysílací frekvenci), což je výhoda pro více zařízení–minimální rušení. Od Bluetooth chceme malou cenu a nízkou spotřebu. Výkonové třídy

- Class 1
 - Maximální výkon 100mW (20dBm)
 - Dosah až 100 m
- Class 2
 - Maximální výkon 2,5 mW (4dBm)
 - Dosah 10 metrů

- Class 3
 - Maximální vysílací výkon 1 mW (0dBm)
 - Dosah 1 metr

Obecně můžeme standardy Bluetooth rozdělit na ty, co jsou zaměřené na přenos dat a ty druhé, které jsou zaměřené na minimalizaci spotřeby. Z tohoto vyplývají 3 základní standardy Bluetooth.

9.2 Bluetooth Basic Rate/Enhanced Data Rate

Toto je původní verze, které jsou určeny pro souvislý přenos dat na krátkou vzdálenost. Používalo se to ke službám jako streamování audia nebo přenosu dat. Tato verze umí pouze point—to—point spojení. Využívá frequency hoppingu (1600 skoků za sekundu). Má 79 1 MHz kanálů.

9.3 Bluetooth Low Energy

Bluetooth 4.0. Umožňuje vysílat krátké bursty dat na dlouhé vzdálenosti. Výhodné pro apliakce, kde nepotřebuji kontinuální spojení ⇒ IoT aplikace. Objevuje se zde Whitelist (ignoruju všechnz, kdo nesjou na whitelistu). Maximální vysílací výkon je 10mW. Zabezpečení 128bit AES. U BLE jsou kanály přímo rezervované na vyhledávání–rychlejší než u klasického Bluetooth. Využívá 40 2 MHz kanálů a používá Gaussian frequency shift modulaci (GMSK). Stejně jako Bluetooth Classic používá frequency hopping (ale není to úplně stejné, liší se v detailech). Rychlsot u BLE je 1–2 Mbps. Mnohem rychleji se probudí než klasický bluetooth.

9.4 Alternate MAC/PHY (AMP)

Bluetooth 3.0. Pro sestavení se použije klasický bluetooth BR/EDR, samotný přenos pak probíhá přes WiFi.