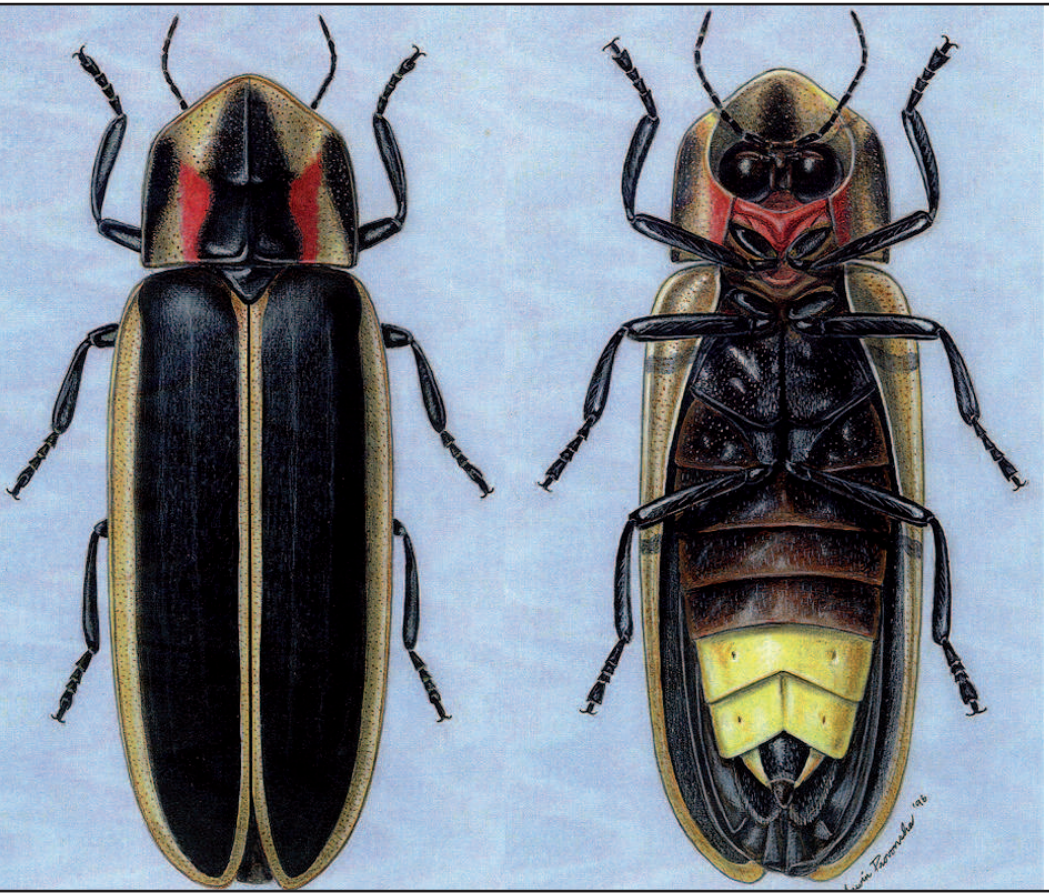
Op tijd voor energie

Intreerede

prof.dr. K. (Koen) G. Langendoen



30 januari 2009

TU Delft

Technische Universiteit Delft

Faculteit Elektrotechniek, Wiskunde en Informatica

Op tijd voor energie

Intreerede

Uitgesproken op 30 januari 2009
ter gelegenheid van de aanvaarding
van het ambt van hoogleraar Embedded Software
aan de Faculteit Elektrotechniek, Wiskunde en Informatica
van de Technische Universiteit Delft

door

prof. dr. K. (Koen) G. Langendoen

Meneer de rector magnificus,
Leden van het college van bestuur,
Collegae hoogleraren, studenten en
andere leden van de universitaire gemeenschap,
Zeer gewaardeerde toehoorders,
Dames en heren, jongens en meisjes,

Inleiding

Al doende leert men. Dat is het motto dat ten grondslag ligt aan het onderzoek en het onderwijs dat ik hier verzorg aan de Technische Universiteit Delft. Beide aspecten, onderwijs en onderzoek, zullen in de komende rede uitgebreid aan bod komen.

Onderwijs

Laat ik met het onderwijs beginnen. Sinds enige jaren waait er een frisse wind door onderwijsland in Delft en die is geïnitieerd door het college van bestuur middels de notitie "focus op onderwijs", waarin het belang van onderwijs nog eens extra benadrukt wordt. Dat zijn hele mooie woorden uiteraard, maar wat betekent dat nu in de praktijk? In onze faculteit Elektrotechniek, Wiskunde en Informatica betekent dat onder andere dat de Basis Kwalificatie Onderwijs (BKO) is ingevoerd. Elke nieuwe docent, waaronder ikzelf destijds ook, wordt geacht deze kwalificatie te behalen zodat hij didactisch geschoold les kan geven. Een van de dingen die je leert tijdens het BKO-traject is dat het traditionele model van college geven niet erg effectief is. Met het traditionele model wordt bedoeld het **I/O model** waarbij de **O** staat voor **Output**, en die wordt uiteraard verzorgd door de docent, en de **I** staat voor **Input**, en dat is dan de taak van de student die lekker achterover leunend het onderwijs tot zich kan laten komen. Echter, het blijkt veel effectiever om werkelijk met de materie aan de slag te gaan. Vandaar dat het moderne onderwijs dan

ook gestoeld is op interactieve werkvormen waarbij docent en studenten middels vraag en antwoord met elkaar het onderwijs vorm geven. Zo zijn er interactieve hoorcolleges, werkcolleges, en projectonderwijs; deze laatste twee vormen zullen vandaag niet aan bod komen, en ondanks dat de setting in deze zaal niet optimaal is, zal ik wel proberen te laten zien wat het interactief college betekent in de praktijk. Kortom de boodschap is: het is vandaag niet simpel zitten en luisteren, maar u wordt geacht mee te doen!

Onderzoek

Laat ik me nu dan bezighouden met het onderzoek. Mijn leerstoel is genaamd Embedded Software en houdt zich bezig met de software die in embedded systemen draait. Nu goed, zult u zich afvragen, wat is dan zo'n embedded systeem? De moderne mens die bedenkt dat niet zelf, maar raadpleegt natuurlijk het Internet en in het bijzonder Wikipedia:

> An **embedded system** is a special-purpose computer system designed to perform one or a few dedicated functions, often with real-time computing constraints. It is usually embedded as part of a complete device including hardware and mechanical parts. In contrast, a general-purpose computer, such as a PC, can do many different tasks depending on programming. Embedded systems control many of the common devices in use today.

Dat zijn inderdaad een heleboel mooie woorden bij elkaar, waarvan de strekking misschien niet in een keer duidelijk is. Vaak helpt het als we eens even naar een voorbeeld van een embedded systeem kijken: de auto, het fototoestel, de spelcomputer en natuurlijk de mobiele telefoon. (Mocht u uw mobiele telefoon nog niet uitgezet hebben, dan is dit een mooi moment om dat even te doen.) Kortom, embedded systemen zijn in het alge-

meen computers die gebruikt worden om andere systemen aan te sturen.

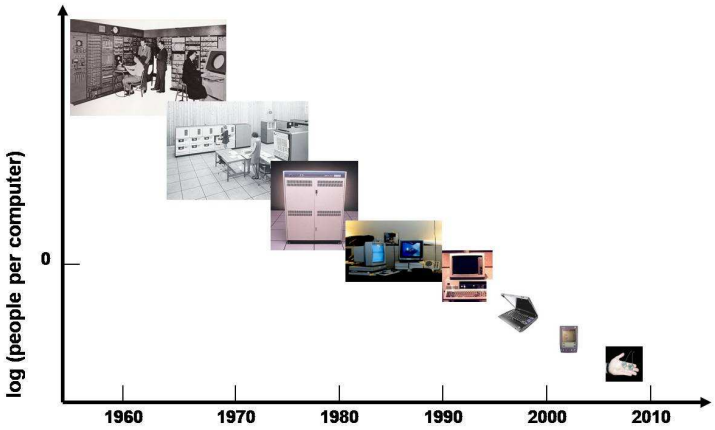
Embedded systemen hebben een aantal eigenschappen die ze interessant maken. Ten eerste zijn ze erg goedkoop, immers consumenten elektronica kan niet al te duur zijn. De computer die er in zit is navenant ook erg goedkoop. Ten tweede worden de meeste embedded systemen gemaakt voor één specifiek doel. Als je de computerchip van een fototoestel overzet in een spelcomputer gebeuren er echt de meest vreemde dingen! Dus embedded systemen worden gemaakt met een specifiek oogmerk. Tot nu toe was het zo dat de meeste embedded systemen gemaakt waren om één ding te doen zoals ik al zei, en ook niets anders konden. Tegenwoordig zie je dat embedded systemen steeds meer met elkaar gaan praten. Moderne fototoestellen hebben een bluetoothchipje aan boord zodat er draadloos gecommuniceerd kan worden met de PC en de digitale foto's gemakkelijk overgezet kunnen worden. Er zijn echter nog veel meer apparaten die bluetooth praten, dus in principe kunnen al die embedded systemen met elkaar communiceren en door samen te werken allerlei nieuwe diensten aanbieden. Daar wil ik nu niet op ingaan, maar het is wel een heel belangrijke tak van onderzoek.

Dan over naar mijn eigen leerstoel Embedded Software. Daar gaat het dus niet over de buitenkant van de embedded systemen, en het gaat ook niet over de hardware, maar het gaat echt over de binnenkant: het programmeren van dat soort systemen. Het inbouwen van een computer is één, maar het apparaat laten doen wat je hebben wilt, dat is twee. Programmeren is een vak op zich en het programmeren van een embedded systeem is eigenlijk programmeren met handicaps want het leven wordt nog een stukje moeilijker gemaakt. Waar komt dat nu door? Die embedded systemen zijn erg goedkoop en dat betekent dat ze ook niet veel kunnen. Er zit weinig geheugen in, de processor loopt niet zo snel, kortom er moet rekening gehouden worden met de beperkte mogelijkheden van zo'n klein embedded computertje.

Verder moeten apparaten direct reageren op de acties van de gebruiker. Zo erger ik me er dood aan als ik op een knop druk en er gebeurt niet meteen iets; een foto moet nú gemaakt, en niet 3 seconden later! Deze realtime eisen vragen natuurlijk een bepaalde manier van programmeren die nu niet meteen in het eerste jaar wordt aangeleerd. Tenslotte is het erg lastig om de software die eenmaal ontwikkeld is voor zo'n apparaat ook werkelijk helemaal foutvrij te krijgen. Je hebt iets in een laboratorium ontwikkeld, je hebt een aantal gebruiksscenario's doorlopen, en vervolgens geef je het apparaat aan de eerste de beste consument die binnen een week al zodanig veel op de knoppen gedrukt heeft dat het apparaat vastgelopen is. Tja, en hoe kom je er nu achter waarom de software vastgelopen is? Het zou heel fijn zijn als je even in dat apparaat kon kijken om te zien wat er met die software gebeurd is. Helaas er zit geen toetsenbord aan, er zit geen beeldscherm aan. Het is dus erg lastig om inzicht te krijgen in hoe het programma executeert. Al die problemen die ik hier genoemd heb kun je als probleem zien maar ook als uitdaging. Dat maakt het programmeren van embedded systemen extra leuk, extra moeilijk.

Wireless sensor networks

Dan wil ik nu overstappen naar het vakgebied waar ik mij de laatste 5 jaar mee beziggehouden heb, dat is het gebied van Wireless Sensor Networks (WSN) en dat wordt gedicteerd door de hardware-ontwikkelingen want als embedded softwareontwikkelaar volg je gewoon de markt. In 1965, dat is een mooi jaar (mijn geboortejaar!), heeft Gordon Moore van Intel reeds geobserveerd dat de miniaturisering van de elektronica volgens een vast patroon verloopt. Zo eens in de 1 à 2 jaar verdubbelt de hoeveelheid transistoren op een chip, en dus ook de performance. Dat is heel mooi, en die trend is dus al decennia gaande. Een



Figuur 1: Bell's law: every decade a new generation

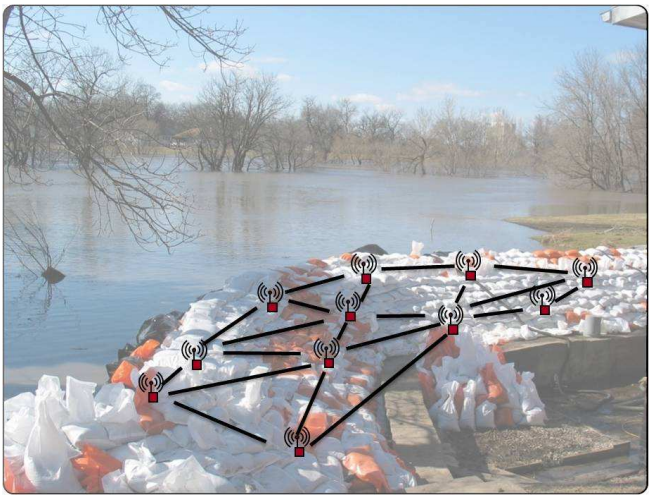
gevolg van deze wetmatigheid is dat computers steeds kleiner geworden zijn, zie figuur 1. We zijn rond 1960 begonnen met computers ter grootte van deze aula, zo rond 1990 was de introductie van de personal computer, en op dit moment zijn we zover dat we allemaal (embedded) computer systemen hebben die we gemakkelijk in onze hand kunnen houden. Figuur 1 geeft ook, op de verticale as, het aantal gebruikers per computer aan. In het begin waren dat natuurlijk vele honderden gebruikers voor die ene grote computer in het rekencentrum. Op het moment dat we bij de personal computer kwamen, was het aantal gebruikers gedaald tot ongeveer één persoon per computer en tegenwoordig, u moet het maar even natellen, zijn er meerder computers per persoon in omloop.

Smart dust

Het is mooi die hardware trend, blijft de vraag "wat moet je ermee als computers extreem klein worden"? Deze vraag, en het

antwoord, markeert het begin van het vakgebied van wireless sensor networks. We spreken dan 1999, dat is ook een mooi jaartal nietwaar Merel? (Merel is geboren op 9/9/99), als een stel Amerikaanse wetenschappers uit Berkeley het "Smart Dust" idee [5] lanceren: in de toekomst worden computers zo klein als stofdeeltjes, en omdat ze kunnen rekenen zijn ze dus nog slim ook! Het briljante van deze Amerikanen was echter niet de handige marketing, maar dat ze bedachten wat je met kleine computertjes kon doen, nl. het combineren van *sensing* (waarnemen), *rekenen* en tenslotte ook nog (draadloos) *communiceren*. Uiteraard moest er ook nog een batterijtje bij voor de stroom. Dat idee van autonome, slimme communicerende sensors, dat is echt het begin geweest van de hele revolutie in sensor networks. Misschien kan ik het beste uitleggen wat je met zo'n klein computertje kunt doen aan de hand van een voorbeeld.

Het voorbeeld geïllustreerd in figuur 2 komt uit het Wisebed project, een Europees project waar mijn onderzoeksgroep in participeert. Eén van de toepassingen waar daar naar gekeken wordt is dijkbewaking, een thema dat zeer relevant is voor Nederland. In dit scenario maken we gebruik van sensoren die vochtigheid kunnen waarnemen, met het idee dat als het ergens erg vochtig wordt we een mogelijk begin van dijkdoorbraak meten, en er dus een alarm gegeven kan worden. Naast de vochtigheidssensor bevat de node ook een processor, zodat er eenvoudige signaalbewerking gedaan kan worden, een radio om de alarm berichten draadloos te kunnen communiceren richting de dijkgraaf, en een batterij om de node te voeden. Eén zo'n sensor node is klein, iets groter dan een euromunt, en dus goedkoop waardoor we er een heleboel kunnen uitzetten op de dijk die bewaakt moet worden. Omdat de capaciteiten van de sensor nodes beperkt zijn, kan een alarm niet rechtstreeks naar de dijkgraaf gestuurd worden, maar zal dat middels een aantal hops door het netwerk moeten gebeuren. Dat maakt de software iets ingewikkelder, maar valt nog wel binnen de capaciteiten van een gemiddelde sensor node.



Figuur 2: Voorbeeld WSN toepassing: dijkbewaking.

In het algemeen kunnen we stellen dat de voordelen van de WSN aanpak zijn, dat er goedkoop en fijnmazig gemeten kan worden, en dat 24 uur per dag in moeilijke omstandigheden waar een mens het al gauw laat af weten.

Prototypes en toepassingen

De wetenschappers onder ons zagen al gauw de potentie van sensor netwerken, en waren de eersten die zijn begonnen het concept toe te passen in hun onderzoek. De eerste prototypes verschenen zo rond 2002 en zijn sindsdien verder ontwikkeld. Ik zal er nu enkele kort toelichten om te laten zien in welke diverse omgevingen sensor networks allemaal toe te passen zijn:

Drenthe Mijn onderzoeksgroep heeft zich samen met wetenschappers uit Wageningen de afgelopen jaren beziggehouden met het doormeten van aardappels op diverse proeflocaties in Drenthe. Hiermee is het microklimaat binnen het

gewas in kaart gebracht zodat er onderzocht kan worden of het zin heeft om de gewasbehandeling te verfijnen tot op plant niveau. Dit project was een leerzame ervaring, en niet alleen op wetenschappelijk gebied: ik weet inmiddels *alles* van aardappels.

Australië Collega's van de UTwente, olv. Paul Havinga, hebben het iets uitdagender aangepakt en zijn afgereisd naar Australië om het Great Barrier Reef door te meten zodat wetenschappers beter inzicht verkrijgen in het aftakelingsproces van het koraalrif.

Zwitserland Ter land, ter zee en in de lucht: collega's van ETH Zürich hebben een sensor netwerk geïnstalleerd bovenop de Matterhorn om metingen te doen aan het natuurlijk verval van rotsformaties onder de uitzonderlijke weerscondities die daar heersen. Deze opstelling draait al enkele maanden succesvol en laat zien dat sensor networks inderdaad onder zware omstandigheden in te zetten zijn.

De wetenschappers waren er dus vlug bij, maar inmiddels zijn er ook enkele commerciële bedrijven actief hier in Nederland met het in de markt zetten van sensor networks. Ik zal nu enkele van deze toepassingen bespreken:

Afvalverwerking Een eerste toepassing komt bij Chess uit Haarlem vandaan en betreft de automatisering van een afvalverwerkingsbedrijf dat oude materialen recyclet. De aangevoerde materialen worden netjes gesorteerd en in verschillende containers gedeponeerd, die uiteraard geleegd moeten worden op het moment dat ze vol raken. Dat betekent een chauffeur waarschuwen die vervolgens die container wegsleept en er een nieuwe voor in de plaats zet. Tot nog toe ging dat altijd met de hand en hing het ervan af of degene die de container volstortte ook inderdaad meldde of de chauffeur moest komen. Nu wordt dat met een

sensor waargenomen en wordt er automatisch een berichtje doorgestuurd op het moment dat de container vol is.

Schilderijbewaking Een tweede toepassing, verrassend genoeg, is het schilderijbewakingssysteem dat geleverd wordt door SOWNet. Elke schilderij wordt uitgerust met een sensor node voorzien van bewegingsmelder. Zodra een schilderij beweegt, door een bezoeker of iets anders, dan gaat er een alarm af. In eerste instantie wordt dat Draadloos doorgezonden naar de portier die dan een suppoost naar de juiste plek kan sturen. Echter mocht dat allemaal niet lukken, omdat de draadloze verbinding gestoord is of er iets anders aan de hand is, dan wordt er een akoestisch alarm afgegeven door de sensor node zelf.

Dan toch weer even terug naar de wetenschap want daar klopt het hart nu eenmaal wat harder van. Sinds kort ben ik met de Universiteit van Amsterdam bezig om vogels te volgen. Dit is geen sinecure aangezien de sensor node die aan de vogel bevestigd wordt extreem licht moet zijn, en de hoge mobiliteit voor allerlei complicaties zorgt. Nu kunt u zich afvragen "wat is de relevantie van het volgen van vogels?", dan breng ik even in herinnering wat er twee weken geleden gebeurde in Amerika, toen er een vliegtuig het pad van een vlucht ganzen kruiste en iets sneller op aarde terugkeerde dan de bedoeling was. De hoop is dat door het (vlieg)gedrag van vogels beter te begrijpen dit soort incidenten in de toekomst vermeden kunnen worden door gericht advies te geven aan de luchtverkeersleiding.

De onderzoeksagenda

De echte wetenschapper heeft altijd een kritische kijk op de zaak en als je met enige afstand naar sensor networks kijkt dan denk je al gauw: Hmm, een flink aantal computers verbonden door

een netwerk ... dat komt wel erg bekend voor, denk Internet. Hmm, draadloze communicatie ... dat kennen we ook: mobiele telefoons. Dus, wat is er eigenlijk nieuw? Deze terechte vraag brengt mij bij de onderzoeksagenda van wireless sensor networks.

Limited resources

Het eerste punt op de agenda is de vraag "hoe om te gaan met de beperkte capaciteiten die sensor nodes bieden". Figuur 3 geeft een overzicht van wat de TNOde's, die we hier in Delft gebruiken bij ons onderzoek, nu werkelijk te bieden hebben. De kenners onder u zien onmiddellijk dat er maar weinig geheugen en processing capaciteit aanwezig zijn. Het belangrijkste echter, en dat is niet in de specificatie te zien, is dat de hoeveelheid energie, die bij zo'n sensor node past, beperkt is. In mijn optiek is de daaruit volgende consequentie, de noodzaak tot energiezuinig opereren, de essentie van het WSN onderzoek, en ook dat wat het onderscheidt van andere disciplines binnen de informatica. Het gaat dus niet om zo snel mogelijk te rekenen, of zo snel mogelijk te communiceren, maar het gaat erom dat te doen op een zo energiezuinigst mogelijke manier.

Robustness

Het tweede agenda punt, en dat wat het programmeren van sensor networks zo moeilijk maakt, is de robuustheid van het hele systeem. Als je je netwerk eenmaal geïnstalleerd hebt op die Matterhorn is het heel vervelend als je na twee dagen terug moet om op het resetknopje te drukken. Dat wil je niet. Dat betekent dus dat de software ontwikkeld in het lab het opeens moet doen op de Matterhorn onder onvoorziene omstandigheden. En dan gaat er nogal eens wat mis, zelfs in een simpel aardappelveld [8]. Draadloze communicatie is bepaald niet storingsvrij. Er kunnen nodes in het netwerk uitvallen, dat kan door falende hardware

ATmega128L CPU (8-bit, 8 MHz)
- 128 KB FLASH (program)
- 4 KB DRAM (data memory)

Chipcon CC1000 radio (868 MHz)
- modulation: FSK 76.8 kBaud
- output power: -20 to 10 dBm

Figuur 3: TNOde met technische specificatie.

maar dat kan ook door software-fouten, en je wilt dat het netwerk zo goed en zo kwaad als dat gaat in ieder geval de data levert die nog op te halen is.

Distributed processing

Het derde agendapunt, en dat is al wat meer richting de toekomst, is dat we aan gedistribueerde processing moeten doen. We hebben allemaal hele kleine, simpele computertjes neergezet. Op zich kunnen ze niet veel, maar je hoopt toch dat ze door samenwerken tot een goede, bruikbare toepassing komen. Vergelijk het maar met een kolonie mieren: elke mier op zich kan niet veel, maar tezamen houden ze toch een heel organisme in stand. Hoe je dat soort gedrag eenvoudig programmeert en dan het gewenste gedrag terugkrijgt, dat staat nog allemaal in de kinderschoenen en daar is nog behoorlijk wat onderzoek nodig om dat in de vingers te krijgen.

Energiezuinigheid

Terugkomen op energie, want ik kan niet op alle onderzoeksaspecten ingaan, laten we maar weer eens naar de hardware kijken: hoeveel energie gaat er in een batterij en wordt het misschien beter? Nou, helaas de wet van Moore (een verdubbeling van

de prestaties elke 1 à 2 jaar) doet geen opgeld in de wereld van de batterijtechnologie. Als je naar de verbetering kijkt over de afgelopen jaren, dan zie je ongeveer een 8 % jaarlijkse groei. Dat betekent dat de batterijcapaciteit slechts eens in de 9 jaar verdubbelt, dus wachten heeft geen zin. Voor de rest kan ik nog opmerken dat de hoeveelheid energie in een batterij beperkt is.

2 kcal (per batterij) 110 kcal (per glas)

Figuur 4: Batterijcapaciteit in relatie tot menselijke maat.

Zoals u aangegeven ziet staan, bevat een batterij ongeveer 2 kilocalorieën. En het moge duidelijk zijn dat dat in het niet valt bij de hoeveelheid energie die wij dagelijks consumeren. Voor de dames dan nog het teleurstellende bericht dat een glaasje wijn helaas nog iets meer energie bevat, zo'n 150 kcal. En voor de kinderen kan ik dan nog mededelen dat een flesje fris gelukkig weer bij de 110 kcal zit.

Vervolgens moeten we de vraag stellen "Waar wordt die energie dan aan besteed?", want als je energiezuinig wilt zijn moet je wel weten waar het verbruik te verminderen valt. Tabel 1 toont het energieverbruik van diverse onderdelen van de TNOde. Links staat de radio, zenden en ontvangen. In het midden staat de processor, die kost ook wel wat energie. En rechts staan de andere onderdelen, de sensoren maar ook de LEDs. De LEDs kunnen we uitzetten, dat is gemakkelijk, net als de processor. Blijft over dat de radio de grote energieverslinder is.

radio (CC1000)		CPU (ATmega128L)		peripherals (sensors/actuators)	
send	18 mA	8 MHz	8 mA	LED	11 mA
receive	12 mA	idle	3 mA	accell.	3 mA
sleep	30 μA	sleep	12 μA	light	1 mA

Tabel 1: Energieverbruik van TNOde componenten.

Laten we maar eens even rekenen op de achterkant van een postzegel: we hebben een batterij, daar zit een hoeveelheid energie in. Ik heb een flinke D-cel genomen (die in een beetje fatsoenlijke zaklamp zit) waar zo'n 7 Ah in zit. Als je de radio aanzet en boodschappen verstuurt of ontvangt dan kost dat ongeveer 20 mA aan vermogen. Als je die twee getallen deelt kom je uit op een levensduur van 360 uur. Dat klinkt wel aantrekkelijk, maar dat is eigenlijk maar gewoon 15 dagen en dat komt dus helemaal niet in de buurt van die 2 of 3 jaar dat we eigenlijk een netwerk in de lucht zouden willen houden. Kortom, we moeten iets.

Duty cycling

Het rekenvoorbeeld laat zien dat we ongeveer een factor 50 à 100 moeten winnen, en dat kan middels een techniek genaamd *duty cycling*. Kort gezegd betekent dit dat je de radio niet constant aan hebt staan, maar dat je hem even aan hebt, dan een tijdje slaapt, dan weer aan zet, enzovoorts. In ons geval zetten we de radio afwisselend voor 1 eenheid aan (active mode) en voor 49 eenheden uit (sleep mode). Dit levert ons de gevraagde factor 50 aan besparing op, maar het kost natuurlijk wel wat. Doordat de radio grotendeels uit staat, wordt de hoeveelheid informatie die je kunt versturen (sterk) beperkt. En het betekent ook dat je af en toe moet wachten voordat er weer zo'n "aan" periode voorbij komt, zodat je kan zenden. Gelukkig kunnen de meeste applicaties in sensor networks die trade-off wel hebben.

Synchronisatie

Ik sprak over duty cycling, complicatie daarbij is natuurlijk dat als je gaat communiceren er twee partijen, de zender en de ontvanger, bij betrokken zijn. Het is wel handig als die twee synchroon hetzelfde aan/uit patroon volgen, want als dat niet gecoördineerd wordt is de kans erg groot dat de ontvanger slaapt (dus niet luistert) als er een bericht naar hem verzonden wordt. Dat is zonde, want verspilling van energie!

Kortom, wil je de truc (techniek) van duty cycling goed benutten dan betekent dat dat zowel zender als ontvanger gesynchroniseerd moeten zijn. Echter we waren in een multihopnetwerk bezig, dus eigenlijk moet het hele netwerk gesynchroniseerd zijn. In principe kan dat. We kunnen klokken gelijkzetten, dat is relatief simpel, maar de generalisatie naar een protocol dat ook eenvoudig in een multihopnetwerk toegepast kan worden is veel lastiger. Vooral omdat we op zoek zijn naar een gedistribueerde oplossing. Simpelweg 1 node (dirigent) de maat laten zwaaien is vragen om problemen: als de dirigent uitvalt, gaat het hele netwerk (orkest) plat. Dat is duidelijk niet de gewenste aanpak gezien de robuustheidseis die we eerder gesteld hebben.

Nu kun je òf eens uitgebreid over tijdsynchronisatie nadenken òf je laten inspireren door, en dat zal mijn moeder plezieren, de natuur! In het bijzonder door de zwermen vuurvliegjes die in het donker synchroon lichtsignalen uitzenden, zie figuur 5. In het bijzonder uitsturen met zijn onderlichaam en op de één of andere manier zijn er hele kolonies waarbij dat gesynchroniseerd plaatsvindt. In het echt gebeurt dit met een duty cycle van ongeveer 50%, maar u kunt zich voorstellen dat we dit ook af kunnen stellen op 1/100. Zo'n vuurvliegje bevat nu niet ongelooflijk veel intelligentie, dus zo'n algoritme zou prima geschikt moeten zijn voor die kleine computertjes in het sensor netwerk. Onze robuustheidseis vraagt echter wel dat het onder alle omstandigheden goed gaat. Eén zo'n wolk vuurvliegjes bekijken dat

is natuurlijk een speciale setting, maar werkt het algoritme ook in alle andere configuraties? Gelukkig zijn er altijd theoreten die daar hard aan gaan "rekenen". In dit geval kan ik een paper aanhalen van de heren Mirollo en Strogatz [9], uit 1990, dus voor informatica begrippen al eeuwen geleden. Deze heren hebben bewezen dat voor n identieke oscillatoren (oftewel van die pulserende vuurvliegjes), die op een willekeurig tijdstip starten (dus zonder hun interne klok te synchroniseren) altijd op hetzelfde schema uitkomen mits men opschuift richting de buurman als deze eerder "af" gaat volgens aanpassingsfunctie f. Met andere woorden, als je nu maar goed luistert naar je buurman en je aanpast, dan is het theoretische resultaat dat het niet uitmaakt hoe je begint, je eindigt altijd in zo'n synchroon pulserend systeem.

Figuur 5: Animatie van een zwerm vuurvliegjes.

Dat was het goede nieuws. Het slechte nieuws is dat er nog wel wat extra, ingewikkelde condities staan en met name deze "f is smooth, monotonically increasing, and concave down" zegt mij niet direct iets. Er zijn ongetwijfeld diverse wijze heren op de eerste rij die dat wel uitgezocht hebben en allemaal weten, maar ik ben meer van de praktische aanpak: laten we maar eens een experiment uitvoeren en zien wat er van komt. En ja, daar heb ik uw hulp bij nodig.

> Er volgt nu een experiment waarbij het publiek een willekeurig getal onder de 5 in gedachten moet nemen, moet

aftellen naar 0, en vervolgens ritmisch moet klappen onderwijl zich aanpassend aan de buurman (M/V). De verwachting is dat na een rommelig begin iedereen in de zaal daadwerkelijk hetzelfde ritme te pakken krijgt.

Dat ging heel goed, experiment geslaagd! Dit was echter een simpel experiment en het blijkt dat als je dat algoritme van die vuurvliegjes, wat dus inderdaad erg goed werkt, over wilt zetten naar sensor nodes, dat er dan nog wel wat addertjes onder het gras zitten:

- Ten eerste hebben we enigszins vals gespeeld in die zin dat we tegelijkertijd konden acteren (geluid maken), en konden waarnemen (horen) of dat dat synchroon ging. Een radio daarentegen die kan je of in zend mode zetten of in ontvangst mode; dat is al complicatie nummer 1.

- Ten tweede hebben we constant geluisterd terwijl ik u verteld had dat we gingen duty cyclen om energie te besparen. Dat maakt het probleem nog eens lastiger want als we slechts af en toe luisteren is er een grote kans dat we die klap van de buurman missen, dus ook daar moet over nagedacht worden.

- Het derde punt dat ik aan wil stippen is dat als je eenmaal die klokken gesynchroniseerd hebt, stel dat dat gelukt is, je ook nog rekening moet houden met het verlopen van de (goedkope) klokken op de afzonderlijke nodes. Deze clock drift speelt een belangrijker rol naarmate er minder gecommuniceerd wordt.

En er is meer!

Communicatie protocollen

In het algemeen is communicatie duur, en er effectief gebruik van maken is de sleutel tot succes. Laat me nog enkele complicerende

factoren aandragen. Allereerst het feit dat radio een broadcast medium is. Als ik zeg "He Allard, wat zit je keurig stil zeg", dan luistert u allemaal mee, dat kost allemaal energie, dat is zonde, want die boodschap was alleen voor Allard. Dit probleem staat bekend onder de naam *overhearing*. Verder luistert u nu constant terwijl er niet altijd iets te horen valt. Dat verspilt ook energie, en staat bekend als *idle listening*. Tenslotte is er nog het punt van de dynamische belasting van het communicatienetwerk dat het algoritmisch nog wat ingewikkelder maakt. Ik heb al verteld dat de data wordt verzameld aan de rand van het veld. Daar gebeurt dus meer, en daarom heb je daar een duty-cycle nodig die wat hoger is dan aan de andere kant van het veld, en dat betekent dat je met een uniforme duty-cycle òf daar energie verspilt òf bij het verzamelpunt te weinig data kan verwerken.

U ziet dus een aantal problemen. Er hebben zich allerlei mensen druk gemaakt om die op te lossen, en er is een hele reeks aan oplossingen bedacht. Die noemen we gezamenlijk ook wel: the MAC alphabet soup [6]. MAC staat voor Media Access Control, en dat betekent: wanneer mag ik de radio aanzetten en een boodschap zenden, zodanig dat we niet met zijn allen tegelijkertijd door elkaar aan het zenden zijn en er dus niets goed overkomt. Er is een hele waslijst aan energiezuinige MAC protocollen, waarvan er twee (Crankshaft [3] en T-MAC [1]) hier aan de TU Delft ontwikkeld zijn. Al die protocollen hebben hun eigen waarde. Van sommige kun je zeggen: die is inderdaad minder goed dan andere, maar de meeste protocollen hebben hun eigen waarde: voor een bepaald soort toepassing doet dit protocol het beter dan de rest en omgekeerd.

Toekomstmuziek

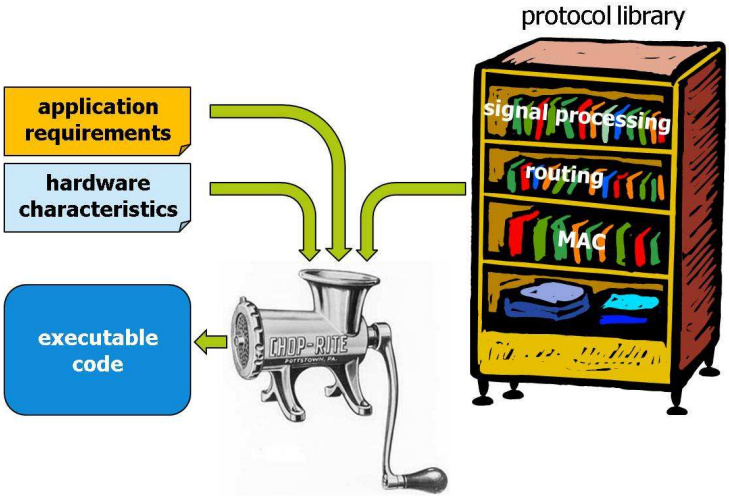
Op dit moment geldt in z'n algemeenheid dat voor een gegeven probleem je goed moet zoeken voor je het juiste protocol hebt.

Dat is op zich een leuke tak van sport, daar kun je hele boekhoofdstukken over volschrijven [2, 4, 7, 10], maar het zou toch fijn zijn als je gegeven een bepaalde toepassing wist welk protocol je het best kon gebruiken. Dat kan, door zo'n boekhoofdstuk na te lezen en erover na te denken. Maar het zou veel fijner zijn als het automatisch zou kunnen, daar zijn tenslotte grote computers voor die dat na kunnen rekenen. En dat is eigenlijk het toekomstbeeld waar ik naar toe zou willen.

Figuur 6 geeft één en ander schematisch weer. Aan de ene kant (het gele blokje) zul je moeten opschrijven wilt je als applicatie. Bijvoorbeeld in geval van de dijkbewakingsapplicatie: we werken met 200 batterijgevoede nodes; we willen wel dat ze minstens 3 jaar meegaan; als er een alarm gerapporteerd moet worden, dan moet dat binnen 5 minuten bij de dijkgraaf zijn; enzovoorts. Dan moet je natuurlijk aangeven welke hardware (het lichtblauwe blokje) er gebruikt gaat worden: we draaien met dit type sensor node, dus er zit zoveel geheugen in; de processor loopt zo snel; de radio kun aan/uit zetten kost zoveel tijd en zoveel stroom. Als je die kennis hebt, schiet het al lekker op.

Aan de andere kant van figuur 6 ziet u eigenlijk een hele kast met standaard oplossingen staan (de protocol library in bruin) met allerlei relevante protocollen, bijvoorbeeld het rijtje met *MAC* protocollen dat we zojuist besproken hebben. Gegeven de applicatie eisen en de hardwarekarakteristieken pak je daar de beste van. Er zit nog meer software in. *Routing* vertelt hoe je door het netwerk van A naar B moet komen. *Signaalbewerking* is essentiëel om de ruis te verwijderen uit de ruwe data die binnenkomt; daar moet je even aan rekenen voordat je weet wat de werkelijke vochtigheid is in het geval van de dijkbewaking.

Als je al die kennis van applicatie, hardware en protocollen bij elkaar stopt, kun je dus de juiste software kiezen; de juiste algoritmes voor MAC, routing en signal processing en dan draai je aan het apparaat en dan komt er hopelijk de kant en klare code uitrollen. Dat zou heel mooi zijn, maar zoiets is er nog niet.



Figuur 6: Toekomstvisie: automatische code generatie.

Gelukkig heb ik wel enig idee hoe we daar zouden kunnen komen. Wat ik voor wil stellen is dat we een modelgebaseerde aanpak nemen. Ten eerste moet je beschrijven wat al die verschillende MAC protocollen wel en niet kunnen. Het is handig als je dat in een wiskundig model verpakt. Als je dan die modellen hebt en die tegen de eisen van de applicatie en de karakteristieken van de hardware legt, kun je dus inderdaad zien: als je deze instelling hebt, krijg je deze performance en dat klopt wel/niet met de eisen. Daar kan aan gerekend worden. Dan kun je dus ook 3 van die modellen op elkaar stapelen en het geheel doorrekenen. Het tweede waarom je een modelgebaseerde aanpak zou willen volgen is dat je eigenlijk tijdens de executie, tijdens het draaien van de software op het sensor netwerk, de instellingen van de protocollen wilt kunnen aanpassen. Dat kan alleen effectief als je weet wat de gevolgen zijn, en daar komt zo'n model dus handig van pas. Stel dat je de duty cycle van een MAC protocol wilt

aanpassen, dan kan je gelijk zien wat dat voor effect heeft op de energiehuishouding, op de hoeveelheid data die door het netwerk gestuurd kan worden, enzovoorts.

Het gebruik van modellen zal dus hopelijk een tweezijdig zwaard opleveren. Aan de ene kant kunnen ze gebruikt worden om software te configureren, aan de andere kant om runtime bij te kunnen sturen in onverwachte situaties. Dit is een toekomstvisie, daar zijn we nog niet. Voorlopig hebben we daar nog allerlei studenten en promovendi voor die ad-hoc bedenken hoe we de software moeten configureren, en op welke manier applicaties zich moeten aanpassen. In de komende jaren echter verwacht ik wel dat we wezenlijke stappen kunnen zetten om die visie in werkelijkheid om te kunnen zetten. Als we daar werkelijk in slagen zou dat iets geweldigs zijn. Misschien zelfs een reden om een eigen bedrijfje te starten. Kortom, dromen genoeg.

Huiswerk

Dat brengt me aan het eind van deze rede; dat betekent niet dat het werk gedaan is: er rest u nog wat huiswerk. Geïnspireerd op de tentamens die ik afneem heb ik hieronder enige juist/onjuist vragen geformuleerd waar u straks tijdens de receptie over kunt discussiëren:

1	op tijd komen spaart energie	juist/onjuist
2	de kredietcrisis is een zegen voor de wetenschap	juist/onjuist
3	elke student verdient een diploma	juist/onjuist
4	kleren maken de man	juist/onjuist

Tenslotte

Graag zou ik nog enige woorden van dank uitspreken. Allereerst richting de faculteit EWI en ook het College van Bestuur voor mijn benoeming en de blijk van vertrouwen die daarmee gepaard

gaat. In het bijzonder wil ik Henk Sips danken voor zijn inspanningen om de nieuwe leerstoel Embedded Software te realiseren binnen onze faculteit.

Ik heb het al de hele tijd over wetenschap gehad. Wetenschap dat doe je niet in je eentje, dat doe je samen met anderen. Daarom wil ik graag alle coauteurs met wie ik de afgelopen 20 jaar artikelen gepubliceerd heb hartelijk danken voor hun bijdrage aan mijn succes.

Als laatste wil ik het thuisfront bedanken. Een wetenschapper is ook maar een mens. Allereerst de kinderen, dat ze me thuis fijn afleiden en hier laten zien dat als het erop aankomt ze zich echt wel kunnen gedragen. Allard, Linde en Merel: Dank jullie wel. En dan wil ik uiteraard mijn vrouw, Gerarda bedanken voor al het onzichtbare werk achter de schermen waardoor ik ongestoord mijn "hobby" wetenschap kan uitoefenen. Simpel gezegd, voor het feit dat ze er altijd is, en beloofd heeft er altijd te zullen zijn! De waardering daarvoor komt maar zelden tot uitdrukking. Eigenlijk hoopt ze dat ik nog eens een boek schrijf, waar ik dan een voorwoord in kan zetten, maar dat zit er de komende jaren denk ik niet in, dus doe ik het maar zo:

"voor Gerarda"

Ik heb gezegd.

Referenties

[1] T. van Dam and K. Langendoen. An adaptive energy-efficient MAC protocol for wireless sensor networks. In *1st ACM Conf. on Embedded Networked Sensor Systems (SenSys 2003)*, pages 171–180, Los Angeles, CA, USA, November 2003.

[2] I. Demirkol, C. Ersoy, and F. Alagoz. MAC protocols for wireless sensor networks: A survey. *IEEE Communications Magazine*, 44(4):115–121, April 2006.

[3] G. Halkes and K. Langendoen. Crankshaft: An energy-efficient MAC-protocol for dense wireless sensor networks. In *4th European conference on Wireless Sensor Networks (EWSN'07)*, pages 228–244, Delft, The Netherlands, January 2007.

[4] K. Kredo II and P. Mohapatra. Medium access control in wireless sensor networks. *Computer Networks*, 51(4):961–994, March 2007.

[5] J. Kahn, R. Katz, and K. Pister. Next Century Challenges: Mobile Networking for "Smart Dust". In *5th ACM/IEEE Int. Conf. on Mobile Computing and Networks (MobiCom '99)*, pages 271–278, Seatle, WA, August 1999.

[6] K. Langendoen. The MAC alphabet soup. `http://www.st.ewi.tudelft.nl/~koen/MACsoup`.

[7] K. Langendoen. Medium access control in wireless sensor networks. In H. Wu and Y. Pan, editors, *Medium Access Control in Wireless Networks*, pages 535–560. Nova Science Publishers, Inc., May 2008.

[8] K. Langendoen, A. Baggio, and O. Visser. Murphy loves potatoes: Experiences from a pilot sensor network deployment in precision agriculture. In *14th Int. Workshop on Parallel and Distributed Real-Time Systems (WPDRTS)*, Rhodes, Greece, April 2006.

[9] R. Mirollo and S. Strogatz. Synchronization of pulse-coupled biological oscillators. *SIAM Journal on Applied Mathematics*, 50(6):1645–1662, 1990.

[10] W. Ye and J. Heidemann. Medium access control in wireless sensor networks. In T. Znati, K. Sivalingam, and C. Raghavendra, editors, *Wireless Sensor Networks*, pages 73–91. Kluwer Academic Publishers, 2004.