Hoofdstuk 1

Resultaten

In dit hoofdstuk zal het ASIC ontwerp van de schakeling die in Hoofdstuk **??** beschreven werd van naderbij bestudeerd worden. Daarbij zal gekeken worden naar de oppervlakte van de schakeling, het verbruik en de maximum bereikbare kloksnelheid $f_{\max}$. Er zal onderzocht worden wat het effect van de verschillende voorgestelde optimalisaties is op al deze parameters. Ten slotte zal het ontwerp vergeleken worden met reeds bestaande implementaties.

Het ontwerp werd geprogrammeerd in GEZEL [14]. Simulaties en compilatie naar VHDL werden uitgevoerd met GEZEL 2.0. De optimalisaties werden doorgevoerd in de VHDL code, aangezien GEZEL dit niet toelaat. Alle ontwerpen werden gesynthetiseerd met behulp van Synopsys Design Vision. De gebruikte bibliotheek was de $0.13\mu m$ *low leakage* bibliotheek van Faraday Technology [4]. Het werd de software verboden flip-flops met test ingangen te gebruiken. De maximale oppervlakte werd ingesteld op nul, wat als netto effect een resultaat met minimum oppervlakte gaf. Verder werd voor het kloksignaal een frequentie van 10kHz gedefiniëerd.

De grootte van alle resultaten wordt uitgedrukt in gates. Dit laat toe te vergelijken met andere resultaten die in de literatuur terug te vinden zijn.

Voor de resultaten in verband met energieverbruik worden steeds twee waarden gegeven. De eerste waarde, dynamisch verbruik, geeft weer hoeveel vermogen verbruikt wordt door veranderende CMOS in- en uitgangen. De tweede waarde, leakage verbruik, is verbruik dat voorkomt zelfs indien een transistor niet geleidt. De impact hiervan hangt onder meer af van de gebruikte bibliotheek.

Deze beide waarden moeten met een stevige korrel zout genomen worden. Het is voor het synthese programma zeer moeilijk hier een nauwkeurige schatting voor te geven. Zolang twee schakelingen echter met dezelfde software, bibliotheek en parameters werden gesynthetiseerd, zijn relatieve vergelijkingen mogelijk. Stel bijvoorbeeld dat het verbruik van ontwerp A geschat wordt op $200nW$ en dit van ontwerp B op $100nW$. Indien er dan voldaan is aan de voorgenoemde voorwaarden, zal het effectief verbruik van B ongeveer de helft zijn van A. Het is dus in het algemeen niet aangeraden vergelijkingen omtrent verbruik te maken met andere bestaande ontwerpen aan de hand van de waarden gegeven in dit hoofdstuk.

Indien gewenst kan het verbruik voor hogere kloksnelheiden geschat worden. Gegeven de standaard frequentie $f = 10$kHz, de formule voor het dynamisch

verbruik van een CMOS schakeling:

$$P_d = V^2 \cdot C \cdot f$$

en het leakage verbruik P_l, kan men dus het totale verbruik omrekenen naar dat van een willekeurige kloksnelheid via

$$P' = \frac{P_d \cdot f'}{10000} + P_l.$$

1.1 Basisimplementatie & register optimalisaties

Het synthetiseren van de meest eenvoudige implementatie, zonder enige optimalisaties aan de registers en met slechts één MALU in de $\mathbb{F}_{2^m}$ kern, geeft de resultaten in Tabel 1.1.

Tabel 1.1: Syntheseresultaten voor de basis implementatie

Opp. (gates)	Verbruik @ 10kHz (nW)		$f_{\max}$ (MHz)
	Dynamisch	Leakage	
31 943	515	134	53.22

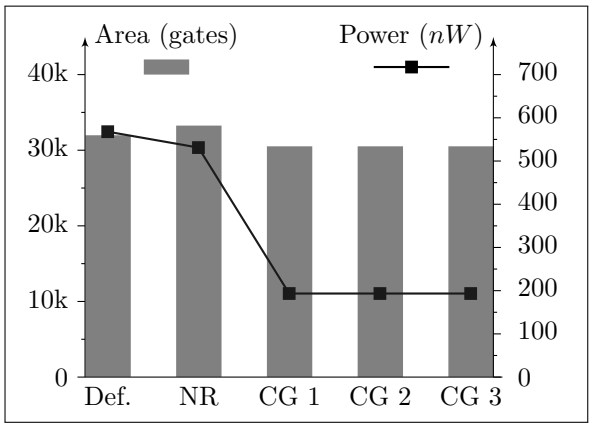
Deze resultaten zullen nu vergeleken worden met de verschillende register optimalisaties. Bij de implementaties van clock gating wordt steeds ook de reset ingangen van zoveel mogelijk registers verwijderd. De synthese resultaten voor de vier verschillende optimalisaties worden gegeven in Tabel 1.2. Zoals verwacht verbruiken al deze resultaten minder dan de niet-geoptimaliseerde versie. De versies met clock gating (CG n) implementeren de schakelingen in de volgorde waarin ze voorkomen in Paragraaf **??**. Voor elke parameter wordt aangegeven hoeveel deze beter is dan in de niet-geoptimaliseerde versie. Ter verduidelijking zijn de oppervlakte en het totale verbruik van deze resultaten ook nog eens uitgezet in Figuur 1.1.

Tabel 1.2: Syntheseresultaten voor de register optimalisaties

	Opp. (gates)		Verbruik @ 10kHz (nW)				$f_{\max}$ (MHz)	
			Dynamisch		Leakage			
Basis	31 943		515		134		53.22	
Geen reset	33 221	1%	473	5%	134	0%	54.08	2%
CG 1	30 481	1%	104	5%	117	5%	47.73	5%
CG 2	30 481	1%	104	5%	117	5%	47.73	5%
CG 3	30 481	1%	104	5%	117	5%	47.73	5%

1.2 Meerdere MALU's

Mits de toevoeging van extra MALU's is het mogelijk de totale rekeningtijd drastisch te verlagen (zie Paragraaf **??**). Hoewel het gebruik van meerdere MALU's de uiteindelijke schakeling vergroot en dat dus enigszins in gaat tegen



Figuur 1.1: Syntheseresultaten voor de basis implementatie met en zonder register optimalisaties

de originele doelstelling, wordt hier toch onderzocht in welke mate de interessante parameters hierdoor juist worden beïnvloed. Er kan dan een afweging gemaakt worden tussen het gebruik van een schakeling met één MALU op hogere kloksnelheid versus één met meerdere MALU's aan een lagere kloksnelheid. De eerste schakeling zal kleiner zijn, maar waarschijnlijk wel meer verbruiken dan de tweede.

De totale berekeningstijd $t = \frac{c}{f}$ kan bepaald worden in functie van het aantal benodigde klokcycli c en het aantal MALU's d:

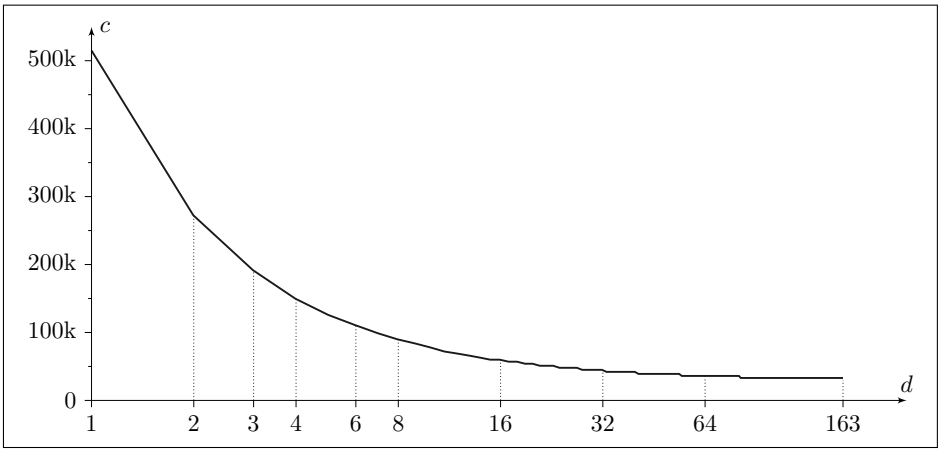
$$c = 27058 + 2993 \cdot \left\lceil \frac{163}{d} \right\rceil,$$

waarbij 2993 het aantal vermenigvuldigingen is dat dient uitgevoerd te worden en de tweede term in de vermenigvuldiging het aantal klokcycli is dat een vermenigvuldiging kost. In Tabel 1.3 wordt voor enkele waarden getoond hoeveel klokcycli nodig zijn om een berekening te voltooien. In Figuur 1.2 wordt hetzelfde weergegeven, maar dan voor elke d van 1 t.e.m. 163. Het is duidelijk dat de tijdsbesparing waar extra MALU's voor zorgen vrij snel teniet wordt gedaan door het aantal cycli dat niet door d beïnvloed wordt.

Tabel 1.3: Aantal klokcycli c nodig voor één pairing i.f.v. aantal MALU's d

d	1	2	3	4	6	8	16	32
c	514 917	272 484	191 673	149 771	110 862	89 911	59 981	45 016

Op de implementaties met meerdere MALU's werden ook steeds de derde clock gating techniek (en het verwijderen van reset ingangen) toegepast, aangezien deze de grootste energiebesparing teweeg brengt. Implementaties met een aantal MALU's gaande van twee t.e.m. tweeëndertig werden gesynthetiseerd. Een nog hoger aantal MALU's zou immers compleet ingaan tegen de originele doelstelling. De resultaten van de synthese zijn te zien in Tabel 1.4 en Figuur 1.3. Indien men nog meer snelheidswinst wenst te boeken, zou het beter



Figuur 1.2: Aantal klokcycli c nodig voor één pairing i.f.v. aantal MALU's d

zijn het gehele ontwerp anders te ontwikkelen (bv. door RAM te gebruiken i.p.v. individuele registers).

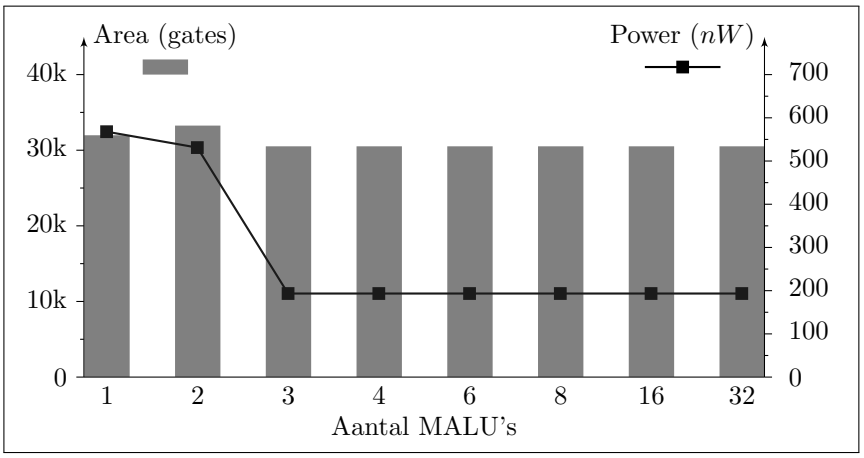
Tabel 1.4: Syntheseresultaten voor meerdere MALU's

d	Opp. (gates)		Verbruik @ 10kHz (nW)				$f_{\max}$ (MHz)		Tijds-winst
			Dynamisch		Leakage				
1	30 481	100%	104	100%	117	5%	47.73	100%	0%
2	30 481	1%	104	5%	117	5%	47.73	5%	47.1%
3	30 481	1%	104	5%	117	5%	47.73	5%	62.8%
4	30 481	1%	104	5%	117	5%	47.73	5%	70.9%
6	30 481	1%	104	5%	117	5%	47.73	5%	78.5%
8	30 481	1%	104	5%	117	5%	47.73	5%	82.5%
16	30 481	1%	104	5%	117	5%	47.73	5%	88.4%
32	30 481	1%	104	5%	117	5%	47.73	5%	91.3%

1.3 Hogere kloksnelheid vs. meerdere MALU's

Aan de gegeven kloksnelheid van 10kHz doet een schakeling met één MALU er 51.5 seconden over om één pairing te berekenen. Dit zal in de meeste gevallen onaanvaardbaar zijn. Daarom wordt hier onderzocht wat de effecten op de schakeling zijn indien de kloksnelheid wordt opgedreven en er eventueel meerdere MALU's gebruikt worden. Aangezien het doel nog steeds blijft de schakeling zo klein mogelijk te maken, zal voor een implementatie met meerdere MALU's enkel die met twee nader onderzocht worden.

Stel een maximale rekentijd $t_{\max} = 50ms$. Voor een implementatie met één MALU moet de klokfrequentie f_1 dan 1030 maal verhoogd worden. Wanneer men de schakeling met twee MALU's even snel wenst te maken als die met één



Figuur 1.3: Syntheseresultaten voor implementaties met meerdere MALU's

dan dient de kloksnelheid f_2 van die eerste vermenigvuldigd te worden met:

$$\begin{aligned}\Delta f &= \frac{272484}{514917} \\ &= 0.52918.\end{aligned}$$

De kloksnelheden van de respectievelijke schakelingen zijn dan:

$$\begin{aligned}f_1 &= 10.3\text{MHz} \\ f_2 &\approx 5.45\text{MHz}.\end{aligned}$$

Ter vergelijking worden de resulterende parameters van beide implementaties gegeven in Tabel 1.5. Beiden werden volledig gehersynthetiseerd met aangepaste parameters voor de klok. De schattingen voor het vermogen zijn dus niet bekomen door de conversie formule aan het begin van dit hoofdstuk toe te passen. Welke van de twee opties de voorkeur zal genieten, zal afhangen van toepassing tot toepassing.

Tabel 1.5: Vergelijking van syntheseresultaten voor twee verschillende implementaties die er even lang over doen één pairing te berekenen

	1 MALU	2 MALU's	
Opp. (gates)	30 481	30 481	120%
f (MHz)	10.3	5.45	52.9%
Verbruik (nW)			
Dynamisch	104	134	118%
Leakage	114	164	144%
$f_{\max}$ (MHz)	47.73	40.12	85%

1.4 Vergelijking met bestaande implementaties

Gezien de vrij recente ontdekking van pairings lag de focus tot nu toe vooral op het snel berekenen van de pairing. Enkele papers beschrijven een ontwerp voor gebruik in sensor netwerken. Helaas hebben deze als doelplatform allemaal een microprocessor. Er werd slechts één paper gevonden waarin het uiteindelijke resultaat naar een ASIC gesynthetiseerd werd. Alle andere recent gepubliceerde implementaties werden ontwikkeld in software of voor FPGA's. Gezien de eerder vermelde focus op snelheid wordt in die gevallen nooit vermeld hoeveel de uiteindelijke schakelingen verbruiken.

Het is dus zo goed als onmogelijk een grondige vergelijking te maken tussen de verschillende bestaande implementaties, gezien hun volledig andere invalshoek. Toch zal zo goed als mogelijk getracht worden enigszins een overzicht te geven, zodat de in deze thesis voorgestelde schakeling beter geplaats kan worden tussen de reeds bestaande ontwerpen.

De ontwerpen specifiek gericht op sensor netwerken worden voorgesteld in [**?**, **?**, **?**]. In alle gevallen wordt de pairing berekend op een ATMega128L microchip. Deze implementaties zijn ontwikkeld voor gebruik op een MICA node, specifiek ontwikkeld voor gebruik in sensor netwerken. Een overzicht van de resultaten is gegeven in Tabel 1.6. Rekening houdend met het stroomverbruik gegeven in [**?**] wordt het verbruik geschat op ongeveer $23.60mW$. Uiteraard zijn de oppervlakte en het verbruik van een microchip implementatie niet te vergelijken met de in deze thesis voorgestelde ASIC schakeling, gezien de zeer verschillende architectuur en de filosifie achter het gebruik van beiden.

Tabel 1.6: Resultaten uit de literatuur voor ontwerpen met focus op sensor netwerken

	NanoECC [**?**]		TinyTate [**?**]	TinyPBC [**?**]
	Binair	Priem		
Veld	$\mathbb{F}_{2^{163}}$	$\mathbb{F}_p$ 160 bit	$\mathbb{F}_p$ 256 bit	$\mathbb{F}_{2^{271}}$
Pairing	Tate	Tate	Tate	η_T
Rekentijd (s)	10.96	17.93	30.21	5.45

In de literatuur zijn vrij veel ontwerpen voor FPGA's terug te vinden. Het probleem is echter dat men zich bij het ontwerp hiervan steeds toelegt op het behalen van een zo hoog mogelijke snelheid, wat resulteert in een grote oppervlakte. Ook wordt vaak gerekend over grote velden (bv. $\mathbb{F}_{2^{313}}$ of $\mathbb{F}_{3^{197}}$), wat meer veiligheid biedt, maar resulteert in nog grotere ontwerpen.

Grootte van een FPGA ontwerp wordt vermeld in slices (voor Xilinx chips) of logical units (voor Altera chips). Deze eenheden zijn onmogelijk om te zetten naar een aantal gates. Zodoende is het dus niet mogelijk de grootte van deze ontwerpen te vergelijken met die van een ASIC schakeling.

Toch wordt in Tabel 1.7 een sumier overzicht gegeven van een zeer beperkt aantal ontwerpen. Bij de selectie hiervan werd vooral gekozen voor ontwerpen waarin in een vrij klein veld gerekend werd. Er dient in acht te worden genomen dat bij al deze implementaties snelheid, en niet oppervlakte, het voornaamste doel is.

Tabel 1.7: Resultaten uit de literatuur voor ontwerpen ontwikkeld voor FPGA's

	Veld	Pairing	Opp. (slices)	f (MHz)	Rekentijd (μs)
Shu *et al.* [**?**]	$\mathbb{F}_{2^{239}}$	Mod. Tate	25287	84	41
Ronan *et al.* [**?**]	$\mathbb{F}_{2^{103}}$	Mod. Tate	21021	51	206
Grabher and Page [**?**]	$\mathbb{F}_{3^{97}}$	Mod. Tate	4481	150	432.3

Bibliografie

[1] G. Bertoni, L. Breveglieri, P. Fragneto, G. Pelosi, and L. Sportiello, "Software implementation of Tate pairing over GF(2^m)," in *DATE 06: Proceedings of the conference on Design, automation and test in Europe*. European Design and Automation Association, 2006, pp. 7–11.

[2] J.-L. Beuchat, N. Brisebarre, J. Detrey, E. Okamoto, and F. Rodrguez-Henrquez, "A Comparison Between Hardware Accelerators for the Modified Tate Pairing over $\mathbb{F}_{2^m}$ and $\mathbb{F}_{3^m}$," in *Lecture Notes in Computer Science*, vol. 5209. Springer, 2008, pp. 297–315.

[3] Certicom Corporation, *SEC 2: Recommended Elliptic Curve Domain Parameters*, september 2000. [Online]. Available: http://www.secg.org

[4] Faraday Technology Corporation, $0.13\mu m$ *Platinum Standard Cell Databook*, 2004. [Online]. Available: http://www.faraday-tech.com

[5] K. Sakiyama, "Secure Design Methodology and Implementation for Embedded Public-key Cryptosystems," Ph.D. dissertation, KU Leuven, december 2007.

[6] L. Batina, N. Mentens, K. Sakiyama, B. Preneel, and I. Verbauwhede, "Low-Cost Elliptic Curve Cryptography for Wireless Sensor Networks," *Security and Privacy in Ad-Hoc and Sensor Networks*, vol. 4357, pp. 6–17, 2006.

[7] D. Hankerson, A. Menezes, and S. Vanstone, *Guide To Elliptic Curve Cryptography*. Springer, 2004.

[8] L. Batina, "Arithmetic And Architectures For Secure Hardware Implementations Of Public-Key Cryptography," Ph.D. dissertation, KU Leuven, 2005.

[9] T. Itoh and S. Tsujii, "A fast algorithm for computing multiplicative inverses in GF(2^m) using normal bases," *Information and Computation*, vol. 78, no. 3, pp. 171–177, 1988.

[10] A. Karatsuba and Y. Ofman, "Multiplication of Multidigit Numbers on Automata," *Soviet Physics Doklady*, vol. 7, p. 595, 1963.

[11] D. Zuras, "On squaring and multiplying large integers," in *Proceedings of IEEE 11th Symposium on Computer Arithmetic ARITH-93*, 1993, p. 260.

[12] Y. K. Lee and I. Verbauwhede, "A Compact Architecture for Montgomery Elliptic Curve Scalar Multiplication Processor," 2006.

[13] M. Müller, A. Wortmann, S. Simon, M. Kugel, and T. Schoenauer, "The impact of clock gating schemes on the power dissipation of synthesizable register files," in *ISCAS (2)*, 2004, pp. 609–612.

[14] P. Schaumont, D. Ching, H. Chan, J. Steensgaard-Madsen, A. Vad-Lorentzen, and E. Simpson, *GEZEL User Manual*, 2007. [Online]. Available: http://rijndael.ece.vt.edu/gezel2