Power Consumption in DFTs
for OFDM Systems
Project Exam - Group 08gr1042

Jes Toft Kristensen
Peter August Simonsen

10th SEMESTER
APPLIED SIGNAL PROCESSING AND IMPLEMENTATION

June 20th, 2008

- Say **hello**
- **Master** presentation, **ASPI**
- **Title**

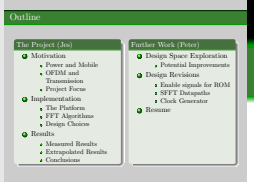
Power Consumption in DFTs for OFDM Systems

Project Exam - Group 08gr1042

Jes Toft Kristensen
Peter August Simonsen

10^{th} SEMESTER
APPLIED SIGNAL PROCESSING AND IMPLEMENTATION

June 20^{th}, 2008



- Present **yourself**
- Tell what you will talk about

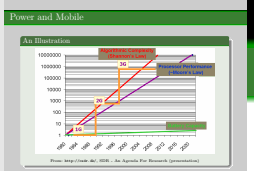
Outline

The Project (Jes)

1. Motivation
   - Power and Mobile
   - OFDM and Transmission
   - Project Focus
2. Implementation
   - The Platform
   - FFT Algorithms
   - Design Choices
3. Results
   - Measured Results
   - Extrapolated Results
   - Conclusions

Further Work (Peter)

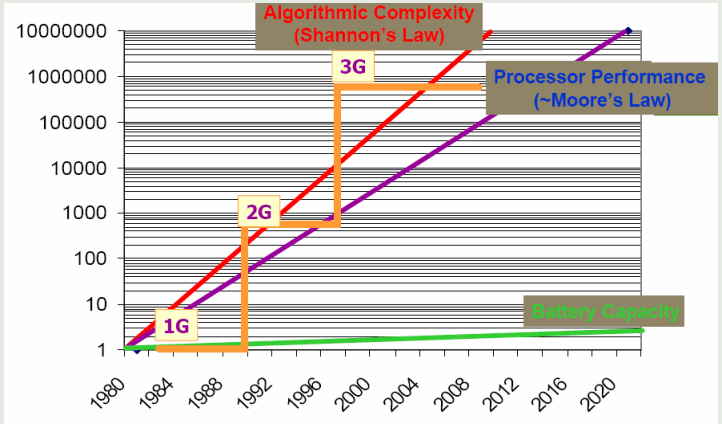
4. Design Space Exploration
   - Potential Improvements
5. Design Revisions
   - Enable signals for ROM
   - SFFT Datapaths
   - Clock Generator
6. Resume



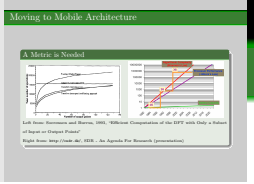
Power and Mobile

- Focus on **mobile applications** (all we talk about happens in phones)
- Processor performance and algorithmic complexity **grows**
- The battery **capacity** does **not follow**
- A great wish is to improve power performance, yields **possibilities** :
  - **Cheaper** battery
  - More **advanced** mobile applications
  - **Longer** battery life
- **Selection** of algorithm for design requires **knowledge of power** usage

An Illustration



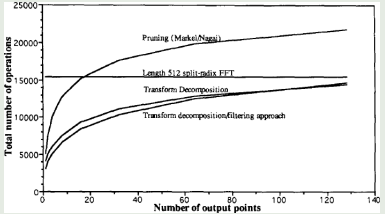
From: http://csdr.dk/, SDR - An Agenda For Research (presentation)

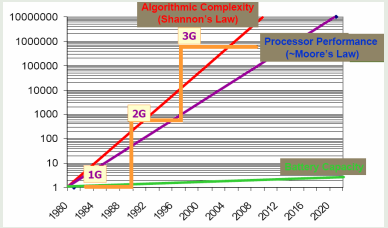


Moving to Mobile Architecture

- A cost metric is needed to make **design choices**
- **Computational complexity** is normally used (Soerensen graph, just an example)
- Is this **comparable** with **power** usage? (Battery capacity)
- If so, can we **save power** by using different algorithms?
- Possibly selecting the **"best"** algorithm from the start
- This is **investigated in this project**

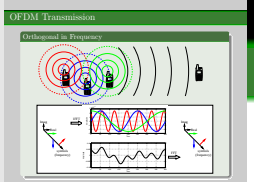
A Metric is Needed





Left from: Soerensen and Burrus, 1993, "Efficient Computation of the DFT with Only a Subset of Input or Output Points"

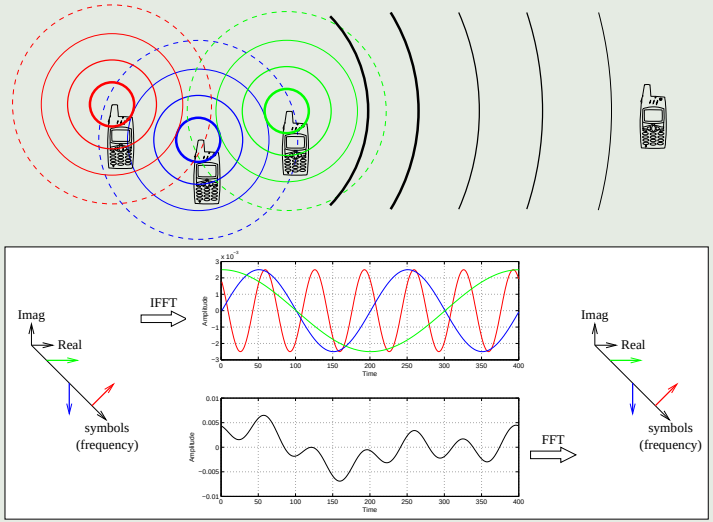
Right from: `http://csdr.dk/`, SDR - An Agenda For Research (presentation)

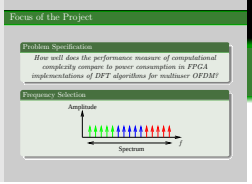


OFDM Transmission

- Transmission **scenario** , sender receivers
- **Separated** by frequency (shown left)
- **Transformed** each symbol **becomes a wave**
- Received mixed on right
- **FFT** at receiver **separates** wave into channels/symbols
- **Select** symbols to compute according to whom you communicate with or channel properties
- Corresponds to a **partial FFT**

Orthogonal in Frequency





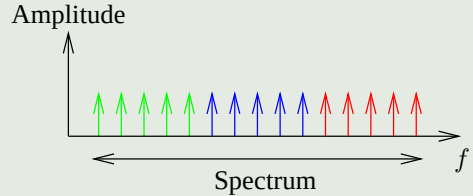
- Say the **problem specification**
- Repeat that this is investigated to see if we can **select algorithms** for low power implementations, based on their **computational complexity**
- Additionally it is investigated if a **frequency selective FFT** is preferable for OFDM, as opposed to a **full range FFT** (lower figure)
- With the **problems defined** , move to **implementation**

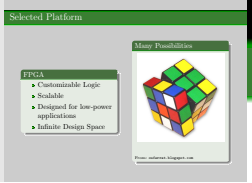
Focus of the Project

Problem Specification

How well does the performance measure of computational complexity compare to power consumption in FPGA implementations of DFT algorithms for multiuser OFDM?

Frequency Selection





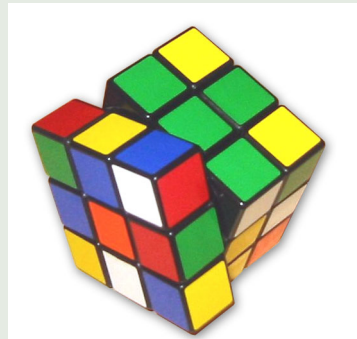
- **Show** the FPGA board
- Development board is the **customizable** platform
- Board is designed for **low-power applications**
- Low-power applications as in **Mobile** systems
- **Infinite** design space
- Leads to **design methodology** , **not covered** here

Selected Platform

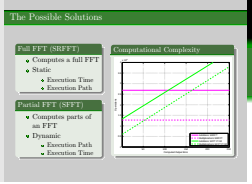
FPGA

- Customizable Logic
- Scalable
- Designed for low-power applications
- Infinite Design Space

Many Possibilities



From: ouferrat.blogspot.com



- **Two** algorithms are selected, one full and one partial
- Split-Radix is a full FFT, very **predictable** . **Low** computational complexity among **full-FFTs**
- Sørensen FFT, also known an **Transform Decomposition** , computes a **subset** of outputs
- Depending on **needed** number of output **bins** , one algorithm is **superior** with regards to computational complexity.
- We must investigate if this also is **true for power consumption?**
- Hence, we make **implementations** of each

The Possible Solutions

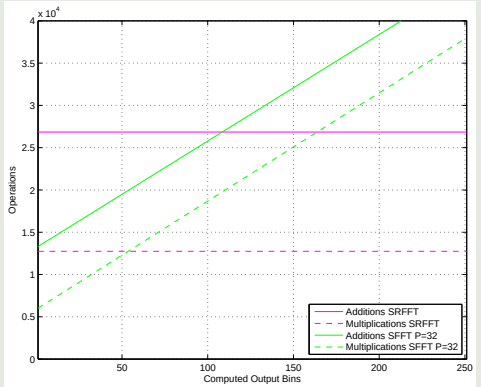
Full FFT (SRFFT)

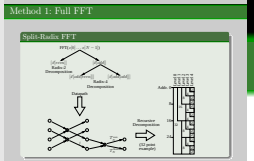
- Computes a full FFT
- Static
  - Execution Time
  - Execution Path

Partial FFT (SFFT)

- Computes parts of an FFT
- Dynamic
  - Execution Path
  - Execution Time

Computational Complexity

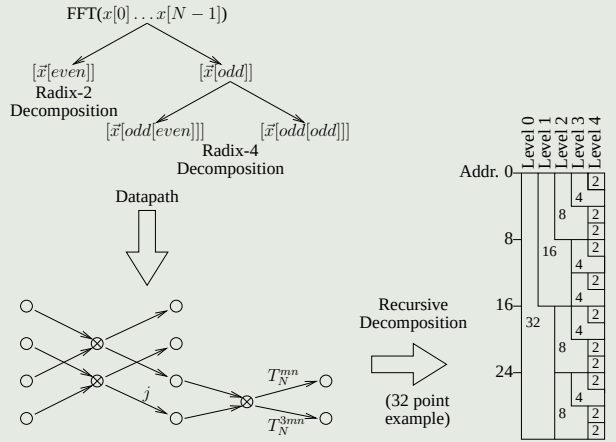


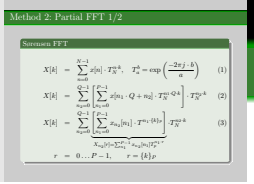


Method 1: Full FFT

- **Rewriting** DFT- **series**
- Split in even and odd
- radix-2 for **even**
- radix-4 for **uneven** saves multiplication
- **L-shaped** datapath
- **Recursive** decomposition
- length: 1024

Split-Radix FFT





- **Rewrite** of DFT-series
- **Innermost** part in (2) is made **independent** of k
- By **modulus** in (3), **periodicity** of twiddle factors
- This is a **DFT** . Calculating a value for **each** r
- **Recombination** is outermost sequence
- The **recombination** step is the **new thing** , everything else is done by the **full-FFT described previously**

Method 2: Partial FFT 1/2

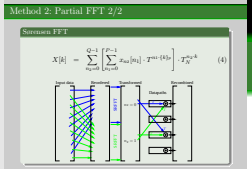
Sørensen FFT

$$X[k] = \sum_{n=0}^{N-1} x[n] \cdot T_N^{n \cdot k}, \quad T_a^b = \exp\left(\frac{-2\pi j \cdot b}{a}\right) \tag{1}$$

$$X[k] = \sum_{n_2=0}^{Q-1} \left[\sum_{n_1=0}^{P-1} x[n_1 \cdot Q + n_2] \cdot T_N^{n_1 \cdot Q \cdot k} \right] \cdot T_N^{n_2 \cdot k} \tag{2}$$

$$X[k] = \sum_{n_2=0}^{Q-1} \underbrace{\left[\sum_{n_1=0}^{P-1} x_{n_2}[n_1] \cdot T^{n_1 \cdot \{k\}_P} \right]}_{X_{n_2}[r] = \sum_{n_1}^{P-1} x_{n_2}[n_1] T_p^{n_1 \cdot r}} \cdot T_N^{n_2 \cdot k} \tag{3}$$

$$r = 0 \ldots P-1, \qquad r = \{k\}_P$$

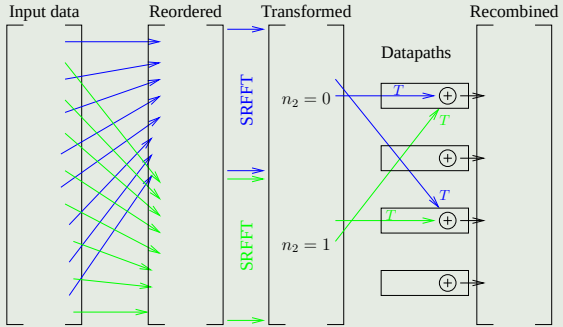


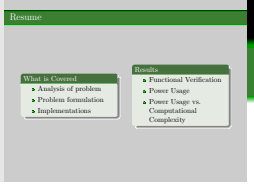
- **Equation** shows smaller FFTs and recombination
- **Innermost** part is steps: input to transformed
- This is done by the **full-FFT**
- **Recombination** is additional **datapaths** (four (4))
- **Design choice** to have 4
- With this in place, we move on to **Results**

Method 2: Partial FFT 2/2

Sørensen FFT

$$X[k] = \sum_{n_2=0}^{Q-1} \left[\sum_{n_1=0}^{P-1} x_{n_2}[n_1] \cdot T^{n_1 \cdot \{k\}_p} \right] \cdot T_N^{n_2 \cdot k} \qquad (4)$$



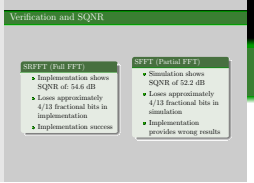


Resume

- We have now examined **analysis, problem formulation and implementations**
  - Problem I: **Power usage in wireless**
  - Proble IIm: **Power Usage vs. Computational Complexity**
  - **Implementations:** Full- and partial FFT
- To be covered:
  - **Functional** verification
  - **Power usage** of algorithms
  - Power **usage** vs. **computational** complexity

What is Covered
- Analysis of problem
- Problem formulation
- Implementations

Results
- Functional Verification
- Power Usage
- Power Usage vs. Computational Complexity



Verification and SQNR

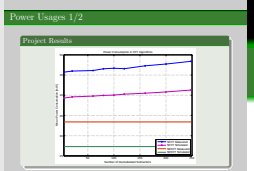
- **Full FFT works** , loses **4/13 fractional bits** in multiplications
- The **partial FFT doesen't work** when when implemented
- **Simulation** shows proper results
- Partial FFT does the **functionally correct thing** , it is thus **safe** to use it for **power measurements**

SRFFT (Full FFT)

- Implementation shows SQNR of: 54.6 dB
- Loses approximately 4/13 fractional bits in implementation
- Implementation success

SFFT (Partial FFT)

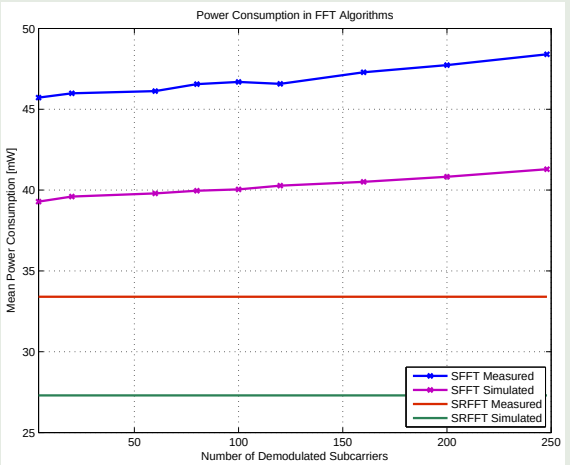
- Simulation shows SQNR of 52.2 dB
- Loses approximately 4/13 fractional bits in simulation
- Implementation provides wrong results

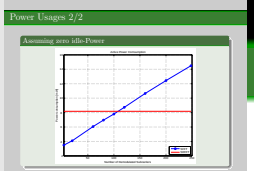


Power Usages 1/2

- **Measured and simulated** results
- **Constant offset**
- Partial FFT is **worse** than Full FFT
- This is **surprising**
- **Static** power is to blame, **always present** if turned on.
- **Even** if not **clocked** or **idle**
- **Regardless** of FPGA utilization (number of logic units used)
- **Results** in high **idle-power usage**

Project Results

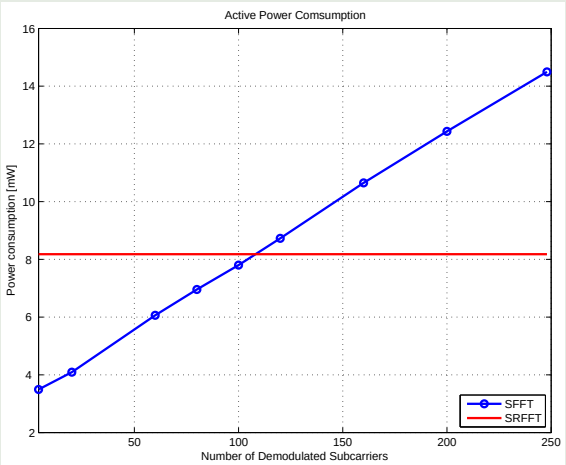


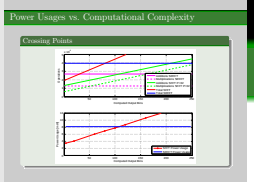


Power Usages 2/2

- **Assume** no power used in idle
- **Biggest impact** on the partial FFT
- **Power saving is possible** for Low L
- It is a **valid assumption**
- Answer to **part of problem specification**
- **Saving** is possible!
- With assumption, it **reminds of computational complexity** (next slide)

Assuming zero idle-Power

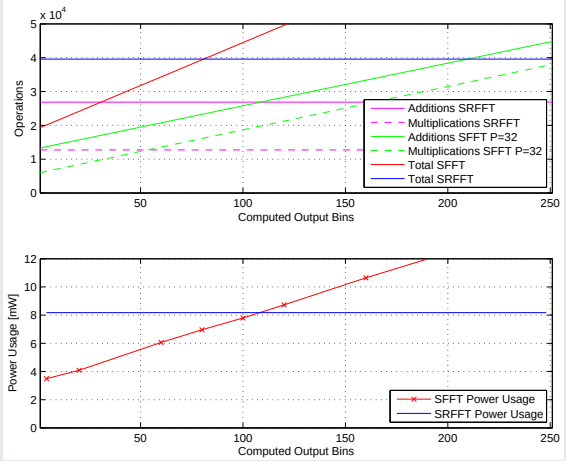


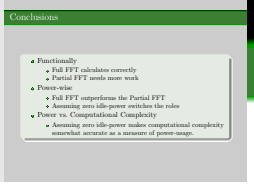


- **Assume** zero idle-Power
- **Upper** is previously shown computational complexity
- **Lower** is measured power usage, with zero-idle assumption
- Compare **crossing points** of computational complexity with that of the measured power.
  - Fits well with **additions**
  - Less accurate with **multiplications**
- **Lump operations together**
- It is a **somewhat indicating** measure for this implementation
- **Better power performance** than expected from computational complexity
- Only applicable for **this case**
- Needs more **study**

Power Usages vs. Computational Complexity

Crossing Points



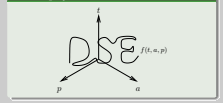


- Full FFT **works**
- Partial FFT **has bugs**
- The Full FFT outperforms the Partial FFT with regards to power
- This changes if the FPGA is turned off during idle
- Computational Complexity is a somewhat indicating measure for this implementation

Conclusions

- Functionally
  - Full FFT calculates correctly
  - Partial FFT needs more work
- Power-wise
  - Full FFT outperforms the Partial FFT
  - Assuming zero idle-power switches the roles
- Power vs. Computational Complexity
  - Assuming zero idle-power makes computational complexity somewhat accurate as a measure of power-usage.

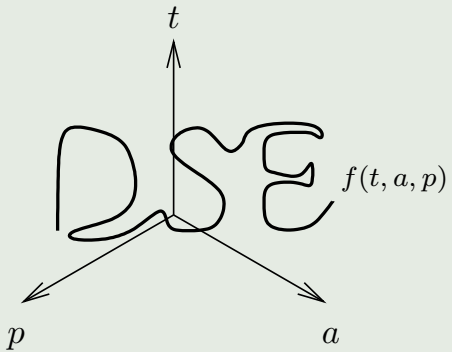
More Exploration?
Infinite Design Space

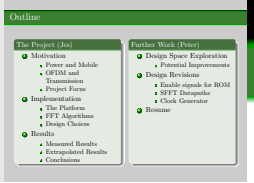


More Exploration?

- Many design choices
- Area, Time, Power
- Manipulate design to achieve maximum of cost-function f
- Peter will examine this

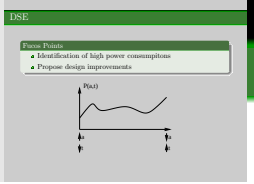
Infinite Design Space





- Present **yourself**
- Tell what you will talk about

Outline

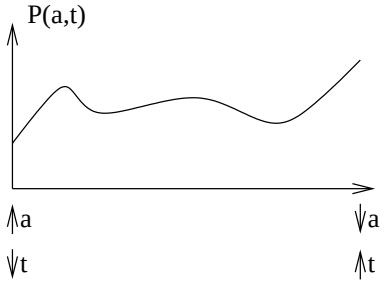


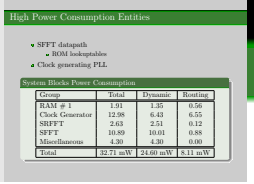
- Purpose of DSE is to **evaluate the design**
- We focus on power **consumption** .
- And thus **rewrite the cost function** to be **power** as a function of **area** and **time**

DSE

Fucos Points

- Identification of high power consumpitons
- Propose design improvements





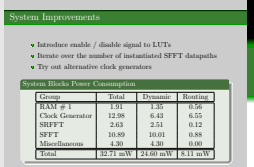
- DSE focuses on final **SFFT with 248 calculated points**
- High power consumptions are found in **clock generator and SFFT datapath**
- Constitutes approx. 24 mW of 32 mW in total
- Data RAM, SRFFT and other use significantly less power.

High Power Consumption Entities

- SFFT datapath
  - ROM lookuptables
- Clock generating PLL

System Blocks Power Consumption

Group	Total	Dynamic	Routing
RAM # 1	1.91	1.35	0.56
Clock Generator	12.98	6.43	6.55
SRFFT	2.63	2.51	0.12
SFFT	10.89	10.01	0.88
Miscellaneous	4.30	4.30	0.00
Total	32.71 mW	24.60 mW	8.11 mW



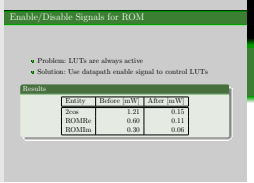
- What can we do to make it better?
- Enable / disable clock to LUTs
- investigate area vs time (or freq) tradeoff
- Disable PLL since they (it) use a lot of power

System Improvements

- Introduce enable / disable signal to LUTs
- Iterate over the number of instantiated SFFT datapaths
- Try out alternative clock generators

System Blocks Power Consumption

Group	Total	Dynamic	Routing
RAM # 1	1.91	1.35	0.56
Clock Generator	12.98	6.43	6.55
SRFFT	2.63	2.51	0.12
SFFT	10.89	10.01	0.88
Miscellaneous	4.30	4.30	0.00
Total	32.71 mW	24.60 mW	8.11 mW



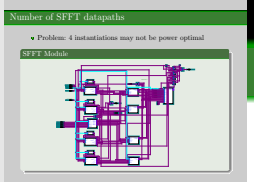
Enable/Disable Signals for ROM

- LUTs in SFFT datapaths consume lots of energy **x 4** .
- Possible source: **Memory is always enabled.**
- Introducing clock enable signals to these blocks **reduce power consumption with up to 87 %** for 2cos ROM

- Problem: LUTs are always active
- Solution: Use datapath enable signal to control LUTs

Results

Entity	Before [mW]	After [mW]
2cos	1.21	0.15
ROMRe	0.60	0.11
ROMIm	0.30	0.06

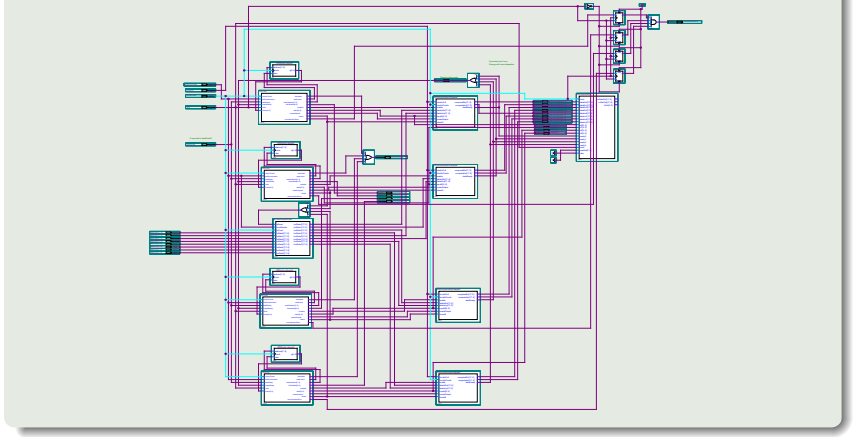


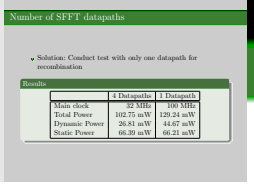
- Suspected high static power consumption from 4 instantiations of SFFT datapaths (uses a lot of area)
- Experiment with low area and high freq.

Number of SFFT datapaths

- Problem: 4 instantiations may not be power optimal

SFFT Module





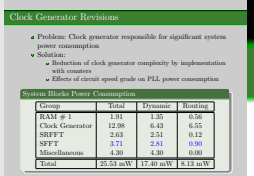
- Notice that **Static power is constant**
- Results suggest that one **should fit the design to utilize the full design**

Number of SFFT datapaths

- Solution: Conduct test with only one datapath for recombination

Results

	4 Datapaths	1 Datapath
Main clock	32 MHz	100 MHz
Total Power	102.75 mW	129.24 mW
Dynamic Power	26.81 mW	44.67 mW
Static Power	66.39 mW	66.21 mW



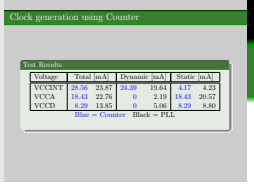
- Purpose of clock generator revision is to investigate the suggested further work on identifying possible power optimizations of the clock generator.
- Work is focused on:
  - Alternative implementation using a counter for clock division (to eliminate use of PLL)
  - Effects of employing different hardware speedgrade settings (since it is possible to change, it might have effects on power consumption...)

Clock Generator Revisions

- Problem: Clock generator responsible for significant system power consumption
- Solution:
  - Reduction of clock generator complexity by implementation with counters
  - Effects of circuit speed grade on PLL power consumption

System Blocks Power Consumption

Group	Total	Dynamic	Routing
RAM # 1	1.91	1.35	0.56
Clock Generator	12.98	6.43	6.55
SRFFT	2.63	2.51	0.12
SFFT	3.71	2.81	0.90
Miscellaneous	4.30	4.30	0.00
Total	25.53 mW	17.40 mW	8.13 mW



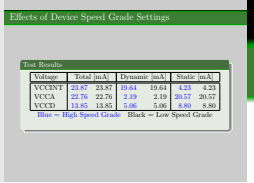
Clock generation using Counter

- Blue **is reference from project**
- Dynamic power consumption is increased in internal logic
  - This i due to movement of where slow clock is generated
- Dynamic power consumption by PLL vanishes.
- Static power consumption of PLLs are almost the same.
  - This **supports the conclusion** from datapaths experiment with regards to **limited power down capabilities** of device.

Test Results

Voltage	Total [mA]		Dynamic [mA]		Static [mA]	
VCCINT	28.56	23.87	24.39	19.64	4.17	4.23
VCCA	18.43	22.76	0	2.19	18.43	20.57
VCCD	8.29	13.85	0	5.06	8.29	8.80

Blue = Counter Black = PLL



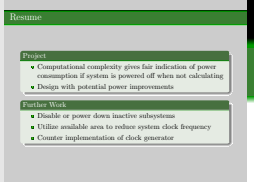
Effects of Device Speed Grade Settings

- Do not spend much time on this slide.
- Changing the device speed grade seems to have no effect powervise.
- So if your design does not require PLLs, get a device without them

Test Results

Voltage	Total [mA]		Dynamic [mA]		Static [mA]	
VCCINT	23.87	23.87	19.64	19.64	4.23	4.23
VCCA	22.76	22.76	2.19	2.19	20.57	20.57
VCCD	13.85	13.85	5.06	5.06	8.80	8.80

Blue = High Speed Grade Black = Low Speed Grade



- Computational complexity is **not far off** with regards to power consumption, when **comparing algorithms** .
  - Requires **outright power down** of inactive circuits
- Static power consumption is a challenge
  - Select device for full utilization
- If PLLs are not needed get device without, and use counters for division

Resume

Project

- Computational complexity gives fair indication of power consumption if system is powered off when not calculating
- Design with potential power improvements

Further Work

- Disable or power down inactive subsystems
- Utilize available area to reduce system clock frequency
- Counter implementation of clock generator

The End
Thank You for Your Time
Questions?

The End

Thank You for Your Time

Questions?