Energy Efficient Communication Links for Smart Devices

Nathan Narevsky
Jan M. Rabaey, Ed.
Vladimir Stojanovic, Ed.
Bruno Olshausen, Ed.

Electrical Engineering and Computer Sciences
University of California, Berkeley

Technical Report No. UCB/EECS-2022-16
http://www2.eecs.berkeley.edu/Pubs/TechRpts/2022/EECS-2022-16.html

May 1, 2022

Acknowledgement

I was lucky to have technical guidance from many senior grad students who I later got to work with as peers, and I would like to thank them. Specifically Dan Yeager, Will Biederman, Chintan Thakkar, Lingkai Kong, Yue Lu, Kwangmo Jung, Matt Spencer, Hanh-Phuc Le, and John Crossley, who taught me a lot about how to design chips and so much more.
I would also like to thank Jackie Leverett, Andrew Townley, Viki Szvortyka, Bonjern Yang, Nick Sutardja, Jaeduk Han, Zhongkai Wang, Stevo Bailey, Woorham Bae, Eric Chang, Antonio Puggelli, Greg Lacaille, and Kosta Trotskovsky for all the work that we did together, and for all of the discussions we had.
I would like to thank Brian Zimmer for being a great mentor to learn from and for making the class that we taught together possible.

Energy Efficient Communication Links for Smart Devices

by

Nathan Narevsky

A dissertation submitted in partial satisfaction of the
requirements for the degree of
Doctor of Philosophy

in

Engineering – Electrical Engineering and Computer Sciences

in the

Graduate Division

of the

University of California, Berkeley

Committee in charge:

Professor Jan Rabaey, Chair
Professor Vladimir Stojanovic, Professor Bruno Olshausen

Spring 2021

Energy Efficient Communication Links for Smart Devices

Abstract

Energy Efficient Communication Links for Smart Devices

by

Nathan Narevsky

Doctor of Philosophy in Engineering – Electrical Engineering and Computer Sciences

University of California, Berkeley

Professor Jan Rabaey, Chair

With the continued development of new sensors, connected devices, and continued scaling data rates there is a serious data problem. This large amount of data needs to be communicated efficiently to where it needs to go for processing, logging or actions. There are many applications where the incoming data that needs to be communicated is not a continuous data rate, the required data rate and latency of the link change over time and settings, so a truly efficient system should be able to take advantage of this burst nature and operate accordingly.

For the purpose of developing larger channel count systems for brain machine interfaces and neuroscience research, a compression algorithm is proposed to minimize the amount of data required to send by up to 700x. This would enable fully wireless systems with larger channel counts, with the main energy bottleneck still being the communication link. A low duty cycle burst mode millimeter wave phased array is proposed to further improve the energy efficiency of the entire system, enabling efficient wireless transfer of a scalable data rate over more than 1 to 100 Mbps. Similar design techniques to this communication link are also applied towards a high speed serial link design, tailored towards a large scale phased array system. The rapid on and off operation allows for a low latency communication interface throughout the entire array without overhead of a constantly operating link. This enhancement reduces the overhead power required to synchronize all of the elements, allowing for a more efficient overall system level design.

Contents

List of Figures

List of Tables

Acknowledgments

I want to thank my advisors and all of the faculty members at the Berkeley Wireless Research Center for all of their guidance and technical discussions. I would also like to thank all of the staff members at BWRC who I have interacted with over the years, the center would likely not function without many of their help. I was lucky to have technical guidance from many senior grad students who I later got to work with as peers, and I would like to thank them. Specifically Dan Yeager, Will Biederman, Chintan Thakkar, Lingkai Kong, Yue Lu, Kwangmo Jung, Matt Spencer, Hanh-Phuc Le, and John Crossley, who taught me a lot about how to design chips and so much more.

I would also like to thank Jackie Leverett, Andrew Townley, Viki Szvortyka, Bonjern Yang, Nick Sutardja, Jaeduk Han, Zhongkai Wang, Stevo Bailey, Woorham Bae, Eric Chang, Antonio Puggelli, Greg Lacaille, and Kosta Trotskovsky for all the work that we did together, and for all of the discussions we had.

I would like to thank Brian Zimmer for not only being a great mentor to learn from, but also for making the class that we taught together possible, I could not have done it without all of the work that he put in.

And lastly I would like to thank my family for everything. Specifically, my parents for their continued support through everything, my sister and brother in law for their help with my final writing push, and my niece and nephew for "helping" with the actual writing.

Chapter 1

Introduction

The amount of data that is generated is rapidly expanding with the large number of interconnected devices being presented everywhere. The IEEE reports data about devices and their trends as part of its International Roadmap for Devices and Systems, or IRDS for short. Looking at the data from this roadmap for 2020 (fig. 1.1), specifically the High Bandwidth Memory (HBM) bandwitch and the per fabric lane speed, the speed of connecting these devices has increased and will continue to to so moving forward.

Table SA-1 Difficult Challenges

	2019	2020	2021	2022	2023	2024	2025	2026	2027	2028	2029	2030	2031	2032	2033	2034
# cores per socket	38	42	46	50	54	58	62	66	70	70	70	70	70	70	70	70
Processor base frequency (for multiple cores together)	3.00	3.10	3.20	3.30	3.40	3.50	3.60	3.70	3.80	3.90	4.00	4.10	4.2	4.3	4.4	4.5
Core vector length	512	512	1024	1024	1024	1024	2048	2048	2048	2048	2048	2048	2048	2048	2048	2048
L1 data cache size (in KB)	36	38	38	40	40	42	42	44	44	44	44	44	44	44	44	44
L1 instruction cache size (in KB)	48	64	64	96	96	128	128	160	160	160	160	160	160	160	160	160
L2 cache size (in MB)	1	1.5	1.5	1.5	2	2	2	2.5	2.5	2.5	2.5	2.5	2.5	2.5	2.5	2.5
LLC cache size (in MB)	67	73	81	89	97	107	118	130	143	157	173	190	200	200	200	200
# of DDR channels	6	8	8	10	10	12	12	12	12	12	16	16	16	16	16	16
HBM ports	4	4	6	6	6	6	6	6	6	6	6	6	6	6	6	6
HBM bandwidth (TB/s)	2.4	2.4	6	6	6.6	6.6	6.6	6.6	6.6	6.6	6.6	6.6	6.6	6.6	6.6	6.6
Fabric lanes	64	72	80	88	96	104	112	120	128	136	144	152	152	152	152	152
Per lane (GT/s)	56	56	56	56	56	56	56	56	56	56	56	100	100	100	100	100
Socket TDP (Watts)	226	237	249	262	275	288	303	318	334	351	368	387	387	387	425	425

L1 = level 1 cache; LLC = last-level cache; Fabric = PCIe or new accelerator fabric (CXL/Gen-Z/openCAPI/CCIX); TDP = total power dissipation.

Figure 1.1: Circuit and System Design Challenges - IRDS 2020 Roadmap

The trend continues as well if we look specifically at mobile and internet of things (IOT) devices as seen in fig. 1.3, looking at the max data rate for 5G and Wi-Fi, and also from the highlighted line as well as the entire SA Mobile Table from fig. 1.2.

Table ES1 Overall Roadmap System Characteristics

2020 IRDS IFT Driver (Exec Summary) Prep - ORSC							
YEAR OF INTRODUCTION	**2019**	**2020**	**2022**	**2025**	**2028**	**2031**	**2034**
Cloud Computing (CC)							
# Cores per Socket [1]	38	42	50	62	70	70	70
Processor Base Frequency (for multiple cores together) [2]	3.00	3.10	3.30	3.60	3.90	4.20	4,5
L1 Data Cache Size (in KB) [3]	36	38	40	42	44	44	44
L1 Instruction Cache Size (in KB) [4]	48	64	96	128	160	160	160
HBM Bandwidth (TB/s) [5]	2.4	2.4	6	6.6	6.6	6.6	6.6
Into-Out of Server Data Rate/lane (Gb/s) (Package) [6]	56	56	56	56	56	100	100
Socket TDP (Watts)	226	237	262	303	351	387	425
SA Mobile Table - Focus Drivers Line Items							
# CPU cores	10	10	12	18	25	28	30
# GPU cores	16	32	32	64	128	256	512
Max Freq (GHz)	2.8	3.0	3.7	4.9	6.5	8.6	11.5
Cellular Data rate (Mb/s)	22	22	1000	1000	1000	1000	1000
5G Maximum Data Rate (Gb/s) [1] - NEW Note added by OSC]	1	5	5	7	10	20	50
# Sensors	6	8	10	12	12	16	16
Board Power (mW)	5096	5351	5899	6829	7906	9152	10594
SA IoT Table - Focus Drivers Line Items							
CPUs per device	1	2	2	4	6	8	8
Max CPU Frequency (MHz)	277	300	310	325	341	357	375
Energy Source (B = battery, H = energy harvesting)	B+H	B+H	B+H	B+H	B+H	B+H	B+H
Sensors per device	4	4	8	12	16	16	16
SA CPS Table - Focus Drivers Line Items							
Number of Devices	64	64	64	128	256	512	512
CPUs per Device	4	4	8	12	12	16	16

Figure 1.2: Overall Roadmap System Characteristics - IRDS 2020 Roadmap

Table SA-2 Mobile Technology Requirements

	2019	2020	2021	2022	2023	2024	2025	2026	2027	2028	2029	2030	2031	2032	2033	2034
# CPU cores	10	10	12	12	18	18	18	25	25	25	28	28	28	30	30	30
# GPU cores	16	32	32	32	64	64	64	128	128	128	256	256	256	512	512	512
Maximum frequency (GHz)	2.8	3.0	3.3	3.7	4.	4.4	4.9	5.3	5.9	6.5	7.1	7.8	8.6	9.5	10.4	11.5
Number of cameras	3	3	3	4	4	4	6	6	6	8	8	8	8	8	8	8
Camera resolution (MP)	12	15	15	18	18	20	20	20	24	24	24	24	24	24	24	24
Number of sensors	6	8	8	10	10	12	12	12	12	12	16	16	16	16	16	16
5G Max data rate (Gb/s)	1	5	5	5	7	7	7	10	10	10	20	20	20	50	50	50
Wi-Fi Max data rate (Gb/s)	5	9.6	30	30	30	30	30	30	30	30	30	30	30	50	50	50
Board power (mW)	5100	5350	5620	5900	6190	6500	6830	7170	7530	7900	8300	8715	9150	9610	10090	10590

Figure 1.3: Mobile Technology Requirements - IRDS 2020 Roadmap

These observations demonstrate the need for approaches for both lowering the energy consumption of the systems sending and creating information, as well as increase the maximum speeds and data rates that these systems are able to operate at. Those innovations are going to come as a result of improved process technology, circuit architectures as well as design algorithms, but these gains are most likely going to come as targeted for their

particular applications and may not be the optimal approach for all systems. This is generally not an issue since the target of a large amount of the industry as well as these studies are focused around the main use cases, which are currently data centers and high performance computing, so the workloads and operating conditions are similar enough that the improvements will be noticed. However, there are a large amount of other systems that people are still building and are required for other systems to build, with the main ones that I will be focusing on moving forward being arrays of radios that are trying to target some of the necessary improvements needed in the wireless and IOT space, as well as specifically enabling the advancement of newer sensor applications targeting brain machine interfaces. These application specific optimizations for various communication blocks are going to be achieved by utilizing the added information about how these specific systems operate and what types of data they are operating on.

As seen in the IRDS 2020 roadmap addressing IOT devices as seen in fig. 1.4, the required power per bit is supposed to decrease at a rapid rate in the near future. At the same time, the standby current is also decreasing while the amount of sensors and the operating speed are targeted to both increase. There are some circuit block and architecture advancements that can be changed to help move towards this goal, but there are still some fundamental limits that cannot be overcome.

Table SA-3 Internet-of-things Edge Technology Requirements

	2019	2020	2021	2022	2023	2024	2025	2026	2027	2028	2029	2030	2031	2032	2033	2034
CPUs per device	1	2	2	2	4	4	4	4	6	6	6	8	8	8	8	8
Maximum CPU frequency (MHz)	257	300	305	310	315	320	325	330	335	340	346	351	360	363	369	375
Energy source (B = battery, H = energy harvesting)	B+H	B+H	B+H	B+H	B+H	B+H	B+H	B+H	B+H	B+H	B+H	B+H	B+H	B+H	B+H	B+H
Tx/Rx power/bit (μW/bit)	0.372096	0.227723	0.139707	0.08571	0.05714	0.038093	0.025396	0.01693	0.011287	0.007525	0.005016	0.003344	0.00223	0.001486	0.001486	0.001486
Battery operation lifetime (months)	6	9	9	9	9	9	12	12	12	12	18	18	18	18	18	18
Deep suspend current (nA)	52	44	38	32	27	23	20	17	14	12	10	9	8	7	7	7
Sensors per device	4	4	8	8	8	12	12	12	16	16	16	16	16	16	16	16

Figure 1.4: Internet of things Edge Technology Requirements - IRDS 2020 Roadmap

There is also a large group working on reducing the power consumption of these types of interfaces for the growing field of sensor devices and IOT, where the systems often rely on a battery or an alternative energy source, so every optimization matters. The communication links are often a large percentage of this overall power budget, so system level enery efficiency is often focused on these blocks. Prior work has shown the possibility of creating a wired link that can rapidly turn itself off and on, enabling a scaling of data rates while maintaining a high level of efficiency [3, 28, 25], and in a later chapter several improvements on these architectures will be discussed. The basic principle here is that dependent on the mode of operation that the entire chip is in the data rate is a known quantity, so the parameters can be determined based on that information and the link can be optimized accordingly.

Throughout the next few chapters a few different projects will be discussed, with the common theme being that they all generate a large amount of data, and need to communicate that data to where it needs to go as efficiently as possible, since in most applications that communication energy is a large percentage of the total system. Specifically, a sensor interface for brain machine interfaces will be discussed in detail, with the main focus being on a compression algorithm developed for this system that enables a very large reduction in the amount of data that needs to be transmitted in this energy constrained system. Depending on the application, a lot of information in these sensors is considered noise, so keeping up a large power overhead to transmit this information is not ideal. Instead, some system level optimization can be done to insure that the required information is being sent properly while putting as little of a burden on the link as possible. For the specific example of a large channel count neural recording interface that is used for a specific application may not need the entire data waveform to function properly, so instead if you were to build the system in a way that it could only send what is required it would be a large reduction in that data rate. If the communication interface connected to that is built in a way that the power is reduced as much as possible when the data rate is reduced there is a very direct impact on the overall system power.

To enable further advancements in research in this field it is desirable to build a system with a larger channel count and wirelessly powered. Building upon the prior work to reduce the size and power of the recording interface the missing piece is a low power wireless communication interface that can handle this data rate. There are a few different options for implementing this link and they will be discussed in the context of a wireless sensor system, and the choices made to enable designing a system that can efficiently operate across the range of data rates that such a system would operate over. Prior work in this space has also demonstrated the possibility of designing those systems with a similar approach as discussed from [14] although there are more calibration and tuning loops that are required to make this system function, especially in a phased array configuration.

Finally another phased array system will be discussed, and the same design techniques will be targeted towards an energy efficient burst mode serial link. The calibration loops and adaptation required will be discussed, as well as many of the circuits are approached in a similar way compared to the wireless link, and similarly are tailored to the specific application at hand. The main building blocks will be compared to the typical approaches for a conventional serial link, and the overall work will be compared to other burst mode serial links for the performance metrics that are desirable for these types of communication interfaces.

Chapter 2

Compression Algorithm

2.1 Neural Recording Interfaces

A neural recording interface is a device that can be implanted or connected to the brain in a way that allows an electrical signal to be detected from the activity from various neurons. There are many different types of these devices, but since the focus here is on the large data rates of these sensor type systems the focus will be on the implantable versions of these devices. The signals that these sensors would pick up are a mixture of many different sources, with frequency content ranging from 60 Hz up to around 8kHz. Typical systems in this range will use an 8-10 bit ADC sampling at 20kHz per channel, so the data rate per channel is relatively low compared to other electrical systems. The main challenge is that there is a need for a large number of these channels operating at the same time, so the data rate ends up being 2-3 orders of magnitude higher than an individual channel simply due to the desired channel count.

The field of neural recording interfaces has had a large amount of advancements over the past few years. As such, most of the systems have already achieved high levels of integration and optimization, with many of them having large channel counts while also having wireless telemetry. Some examples of such systems are [10, 15, 18, 9, 16, 6, 27, 26, 11, 24]. The focus of this section will be on the data reduction for such a system, and how you would be able to enable the integration of a more energy efficient communication system connected to it. The system demonstrated in this chapter was connected using off the shelf components as the main communication medium, but the later chapters discuss how one would be able to design that communication interface itself in a much more energy efficient manner.

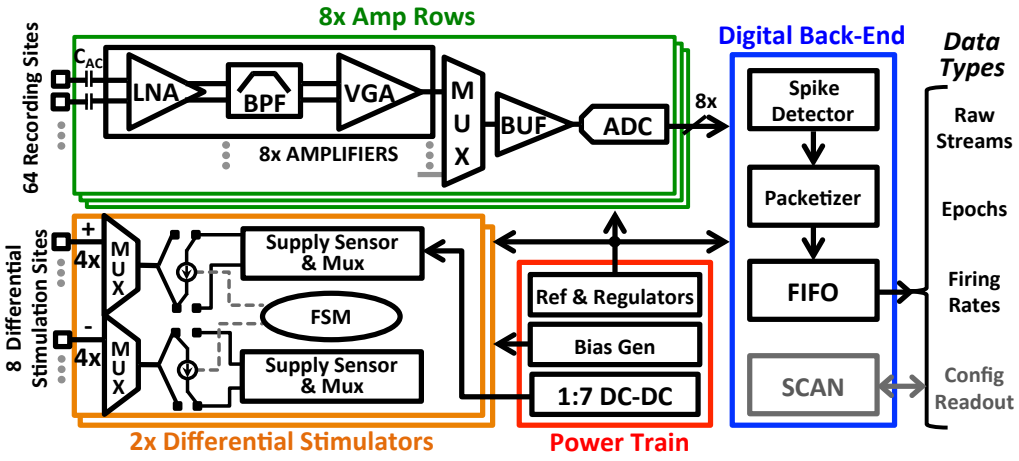


Figure 2.1: Neuralmodulation System Diagram

More details on the overall system can be found in [5], but a brief overview can be found in fig. 2.1. The main elements of this sytem are groups of amplifiers with ADCs, bias generation and power regulation, stimulation, and the digital interface, with programming, readout and packetization.

2.2 Why Compression

High-density implantable recording systems necessitate data reduction, specifically when combined with a data-rate constrained trans-cranial wireless link. Even though the systems are recording biological signals with bandwidths in the range of kilohertz, a system with 64 channels sampling with 10 bits at 20 kilo samples per second generates more than 12 Mbps and necessitates a wireless link with a power budget on the order of 100 microwatts. The data rates generated by these systems are not large compared to modern communication interfaces, such as something more advanced like later generation WiFi links, but the power requirements of such a complicated wireless interface are not possible for these implanted systems. Prior state-of-the-art multichannel neural signal compression implementations [4, 23, 7, 13] require further advances in the level of integration and area-efficiency to achieve a chronically implanted system. In order to make those advancements and enable larger channel count wireless systems with similar levels of system fidelity one necessary improvment is to compress the data that the system needs to send.

It is well studied that most information recorded by neural recording interface systems lies in what are known as spikes, events in the 500 Hz to 3kHz range that are sparse in time. An example of what this waveform looks like, with a typical window of time taken from an in-vivo recording is shown in fig. 2.2. There are a few well known algorithms for trying to pick up these signals from the incoming data, the simplest of which being a simple threshold, potentially after a filter to remove the low frequency signals that are present but not desired

for most applications. More details on this particular algorithm as well as others can be found in [13].

A slightly more sophisticated algorithm known as the nonlinear energy operator (NEO) is frequently used when the threshold has too many false positives, as is also shown in [13]. This algorithm detects the energy in the signal and can be used for higher fidelity for spike detection compared to the thresholding. In applications where that performance is not enough, more complicated algorithms such as singular value decomposition (SVD) or wavelet transforms are used, but these algorithms incur a large increase in computational complexity, which is hard to justify with the power and area constraints of implantable systems.

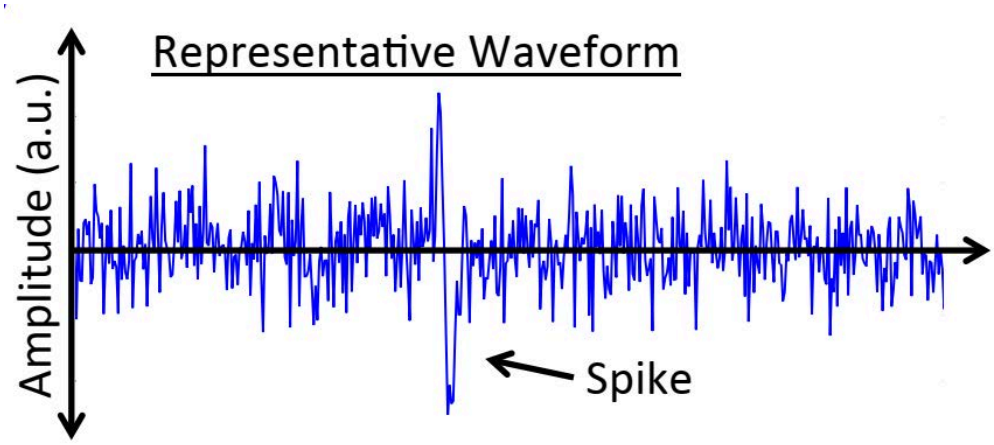


Figure 2.2: Example Waveform showing neural activity

2.3 Algorithm

Conventional NEO implementations detect spike events when $\nu[n] > thres$, where $\nu[n] = x[n]^2 - x[n-1] * x[n+1]$ and x[n] represents an ADC sample at time n. Neural spikes sampled at the typical rate of 20-30kHz have little energy near the Nyquist frequency [12], however, the NEO algorithm is most sensitive to noise perturbations at $x[n]$ relative to $x[n+1]$ and $x[n-1]$ (the Nyquist sample rate). To increase the detector sensitivity to the signal band and reduce sensitivity to out of band noise, a programmable digital delay, T, is added to the algorithm. The detection signal becomes $\psi[n] = x[n]^2 - x[n-T] * x[n+T]$, where T is user-programmable between 1 and 4. For a given spike waveform, there is an optimal T depending on the frequency content of the spike itself. While this is not a linear system due to the squaring of the current sample, a representative analysis can still be explored by inputting a sine wave into the system, and measure the amplitude of the output. The sine wave is then swept across frequency, and plotted with various settings of T, the delay

between samples used in the proposed NEO algorithm, and the results are shown in fig. 2.3 below.

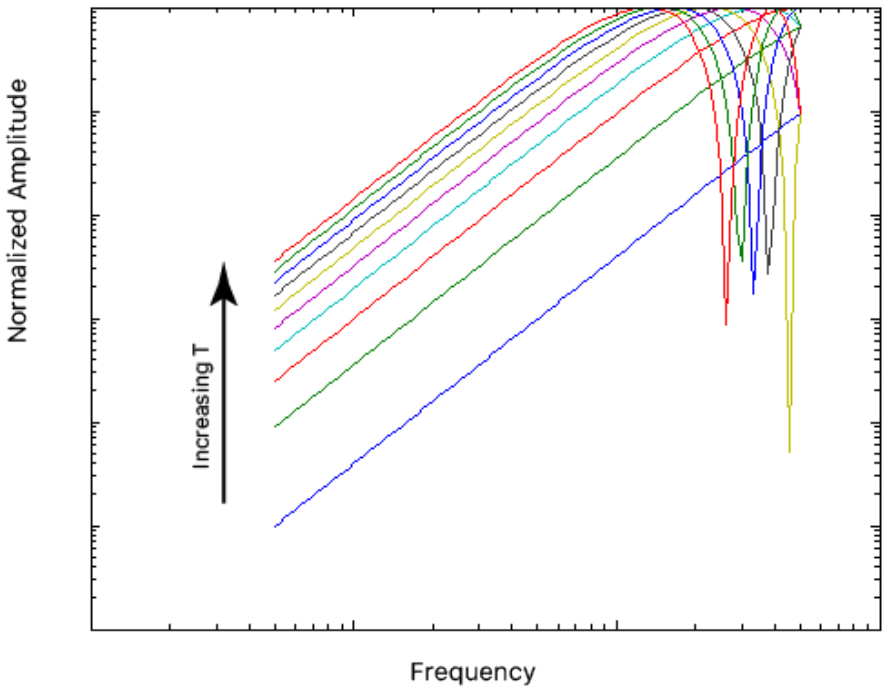


Figure 2.3: Frequency response of the programmable delay NEO

As can be seen from fig. 2.3 as the separation between samples increases, the pole in the frequency response gets lower, allowing this parameter to have some influence on which signals are going to trigger classifying as a spike event. The raw NEO operator, $\psi[n]$, can be influenced by single-sample noise perturbations in addition to spike events, which exhibit energy over several consecutive samples. To further reduce the detector's sensitivity to noise, a 4-point moving average filter calculates $\theta[n] = \sum_{n=0}^{3} \psi[n]$. The result is compared to a threshold, $\theta[n] > thresh$, to detect spike events. Lastly, a programmable refractory period limits the time until the next spike event is declared, which prevents double counting.

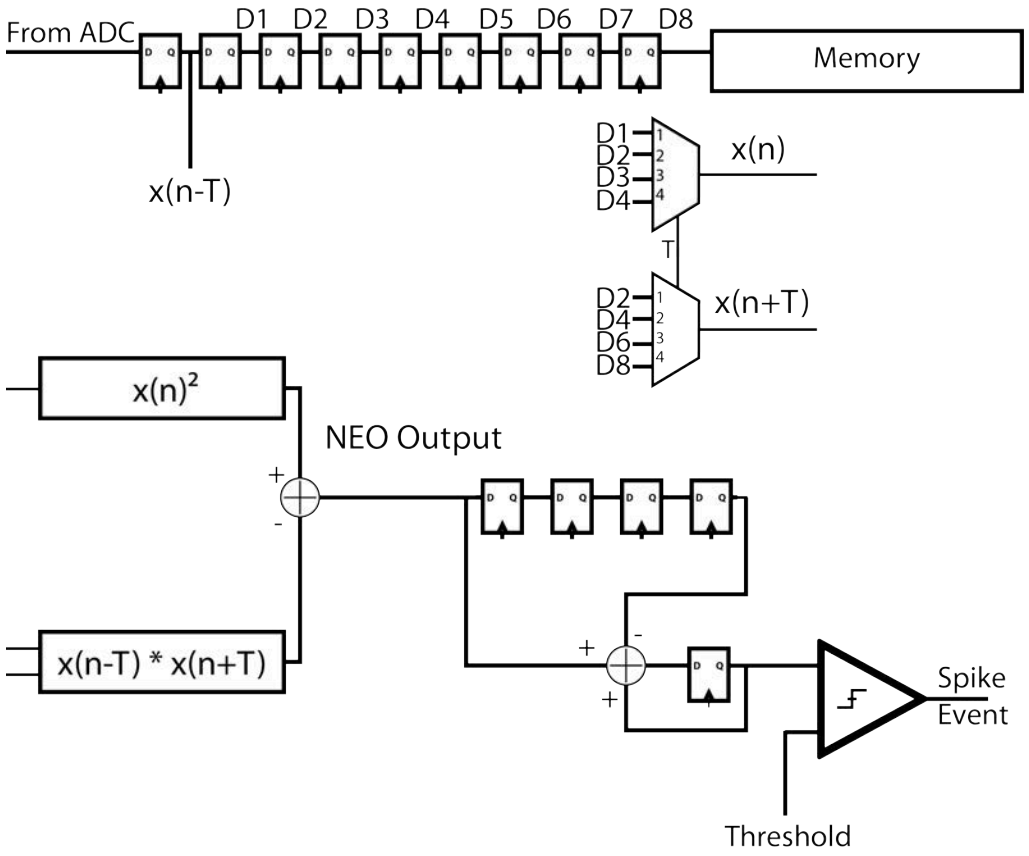
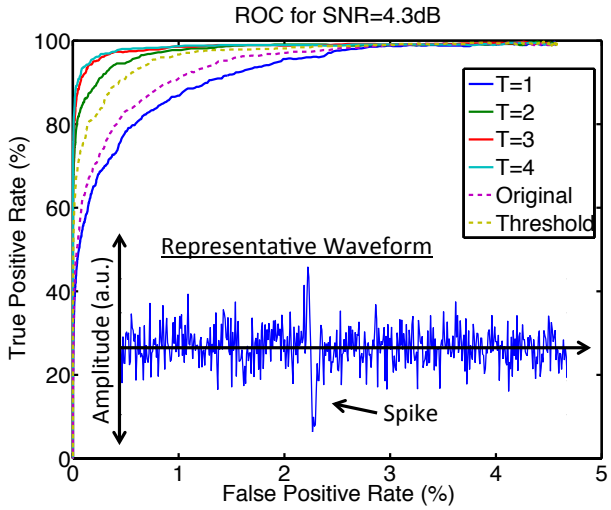


Figure 2.4: Spike detection block diagram showing programmable delay, NEO, and moving average filter.

The algorithm block diagram is shown in Fig. 2.4. Nine stages of 10b registers delay the samples from the ADC, and multiplexers select the sample spacing. Next, the NEO is calculated from $x[n]$ and $x[n \pm T]$ and passed through the moving average filter. Finally, this output is compared to a per-channel user-programmable threshold, which represents detection events. The data is also written into a 48-sample history buffer, which is used to create streaming and epoch data packets. The refractory period is implemented in the packetization block.



Algorithm		Area Under ROC	TP @ FP=.1
Without Filtering	**T=1**	**0.922**	**62.45**
	T=2	0.945	73.77
	T=3	0.951	78.56
	T=4	0.954	79.76
With Filtering	T=1	0.913	55.65
	T=2	0.972	85.22
	T=3	0.977	91.61
	T=4	**0.976**	**93.48**
Threshold		0.963	75.50

Figure 2.5: ROC curve for the proposed algorithm for different values of T (includes filtering), conventional NEO algorithm ("Original") and absolute value threshold ("Threshold").

Fig. 2.5 presents the receiver operating characteristic (ROC) curve generated by varying the threshold of the detector, which illustrates the tradeoff between the false positive (FP) rate and the true positive (TP) rate for a dataset with SNR of 4.3dB. Here, SNR is defined as the ratio between the power of a spike window that only contains a spike and the power in the same window of time that does not contain a spike, averaged over the entire trace. A representative subset of the time domain waveform used for the algorithm is also shown in Fig. 2.5.

The FP rate represents the number of samples incorrectly declared to be a spike, divided by the total number of samples in the trace that are not a spike. This means that a false positive rate of 0.1% would translate into a firing rate of 20Hz given a 20kHz sample rate. The maximum FP rate is 4.75% because the algorithm has a hysteresis time of 1ms (1000 possible spikes per second, or 5%), and the spike firing rate is 50 Hz (50 spikes per second, or 0.25%). For a 0.1% FP rate, the proposed algorithm improves detection rates to 93.5% versus 75.5% for threshold and 62.5% for conventional NEO.

The area under the ROC curve, shown in Fig. 2.5, provides a second measure of comparison between algorithms. The proposed algorithm achieves the best results for T=3 and T=4 spacings. Altogether, these results demonstrate the benefits of both the NEO $x[n \pm T]$ spacing as well as the moving average filter implemented in this algorithm.

2.4 System Implementation

Buffers containing entire windows of time around a spike event ("epochs") are stored in a 1024x32 bit SRAM for all 64 channels. This amounts to 48 10-bit samples per channel,

or 2.4ms of time. Here, SRAM was utilized instead of flops to optimize area and power consumption. This buffer doubles as keeping history on a per-channel basis for the streaming modes, since one streaming packet contains 48 samples from a specific channel. This enables the system to be fully configurable, since everything is controlled on a per-channel basis. These blocks are clock-gated when unused, further optimizing system performance in different modes of operation.

Fig. 2.6 shows the block diagram of the implemented digital back-end, including all signal interfaces and I/Os. A spike detection algorithm based on the nonlinear energy operator (NEO [13]) extracts spike events, enabling data reduction by only sending a 2.1ms time window of data around an event (epochs), and/or spike counts in a 2.4-50ms programmable window. Sending epochs, spike rates, and uncompressed data (streams) can be enabled on a per channel basis via scan instructions. Finally, all packets are put into a clock domain crossing FIFO, which allows the system clock to operate at a different frequency than the output data rate, resulting in further power savings and system optimization.

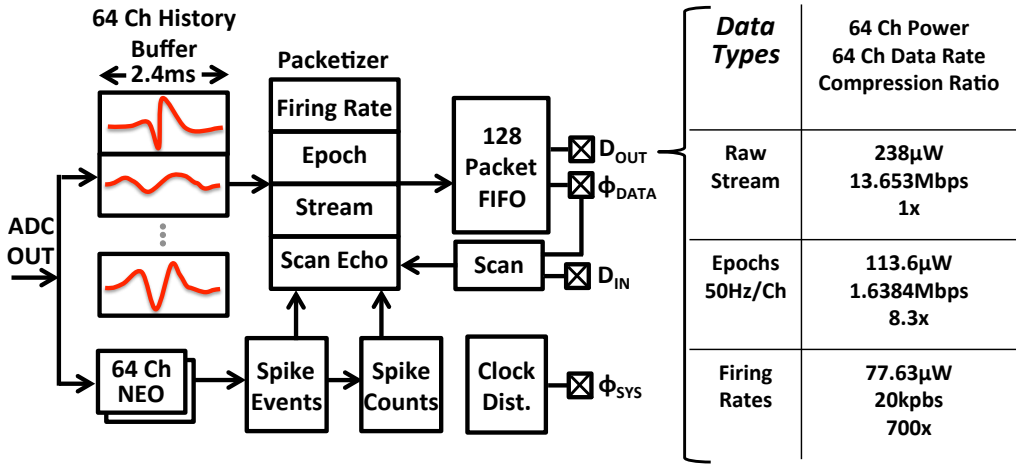


Figure 2.6: Compression block diagram and the power consumption, data rates, and compression ratio for different output modes, with data from the implemented design.

As shown in Fig. 2.6, packets are stored in a 1024x64 bit SRAM FIFO. This results in a buffering capacity of 128 packets (512 bits each). The arbitration for writing into the FIFO gives priority to streaming packets because the raw stream is a superset of the other data representations. Other packet types are written into the FIFO at convenient times when there is space; if there is no room available those packets are simply dropped. Input scan packets are echoed back through the FIFO. This provides acknowledgment and verification, which is essential in a wireless link.

2.5 Performance

The 64 channel digital back-end occupies a total area of 0.675mm^2, a 2.95X reduction compared to previous state of the art [13] normalized to a 65nm technology, including the area for 96kb of memory for channel buffering and a FIFO. Fig. 2.6 summarizes the available data representations and annotates the measured power consumption, data rate, and compression ratio with an average firing rate of 50 Hz on each channel. Firing rates, which are sufficient for BMI control [29], provide the highest compression ratio of 700x. With an average firing rate of 50Hz per channel the total digital power is 77.63µW for firing rates and 113.6µW for epochs, which is lower power than [4]. *In vivo* data demonstrating operation of the compression algorithm as well as illustrating the different data representations is presented in Section 2.6.

2.6 In Vivo System Measurements

A diagram of the testing system designed to seamlessly obtain *in vivo* data is displayed in Fig. 2.7, which includes a compact 0.65" x 0.8" headstage, a base station, and a Graphical User Interface (GUI). The SoC was incorporated onto the headstage, which was designed to sit atop a small animal's head and connect to an implant in the brain. Information is transferred between the headstage and the base station via a 2.6mm diameter μHDMI cable using Low Voltage Differential Signaling (LVDS) for high speed communication. The base station serves as an intermediary between the headstage and the computer's GUI. From the GUI, the user can select which channel(s) to record, as well as send stimulation commands and adjust compression levels on a per channel basis.

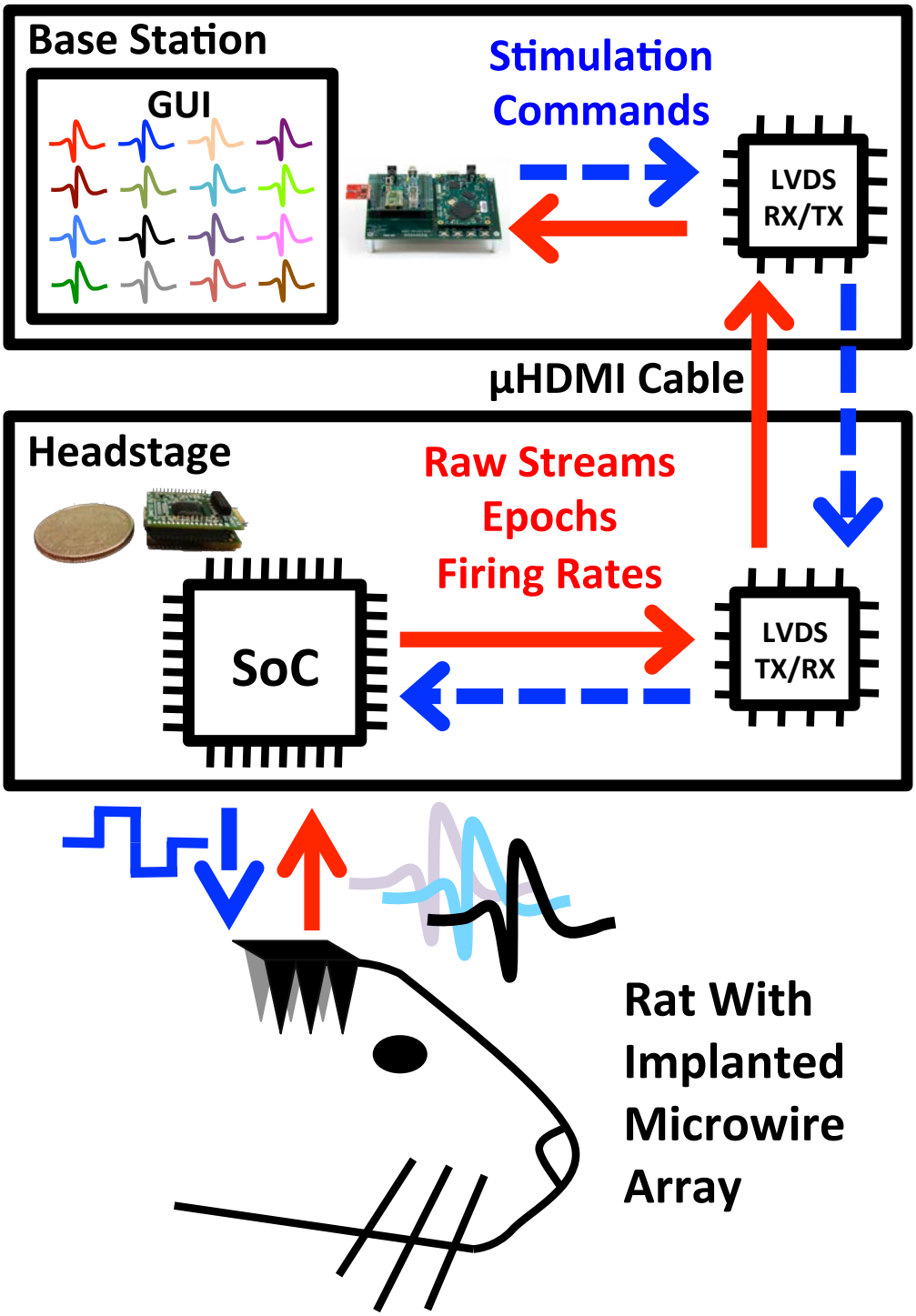


Figure 2.7: The *in vivo* neuromodulation test system is composed of a microwire implanted array, a compact headstage containing the SoC, a base station, and a Graphical User Interface (GUI).

Extracellular recordings were performed using a 16-channel microwire array implanted in the visual cortex of an adult Long-Evans rat. Arrays consisted of teflon-coated tungsten microwires (35µm diameter, 250µm electrode spacing, 250µm row spacing; Innovative Neurophysiology, Inc., Durham, NC, USA). All animal procedures were approved by the UC Berkeley Animal Care and Use Committee. Extracellular recordings were performed for several consecutive days, more than one month after the surgery. Clearly identified waveforms with a high signal-to-noise ratio were chosen for further investigation as single unit responses. Putative single units were validated based on waveform shape, reproducibility, amplitude, and duration. We also verified that the characteristics of the inter-spike interval distributions were close to Poisson and exhibited a clear absolute refractory period.

A typical subset of recorded *in vivo* data is shown in Fig. 2.8, which displays time-aligned epochs recorded from one channel. In order to verify *in vivo* compression accuracy, all three forms of the SoC's outputs were aligned in time, as displayed in Fig. 2.9. Each epoch data packet includes a time stamp, which allows for spike detection confirmation when superimposed onto the raw data stream. In addition, accurate firing rate calculations were verified by ensuring that the firing rate counter incremented with each spike event. As described previously, the SoC computes firing rates over a specified window of time, which in this case was 26.2ms.

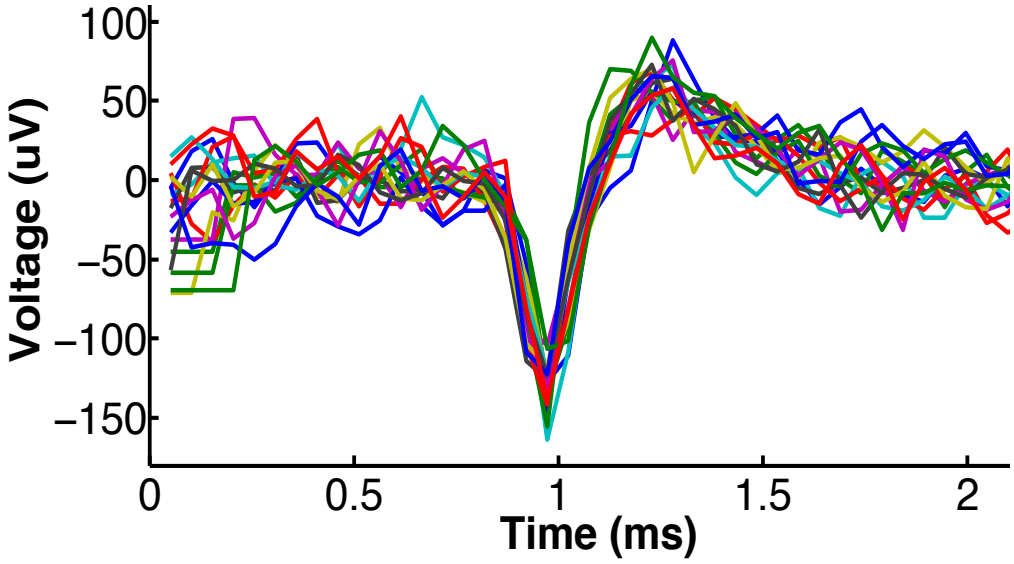


Figure 2.8: Time-aligned epochs recorded from one channel of *in vivo* neural data.

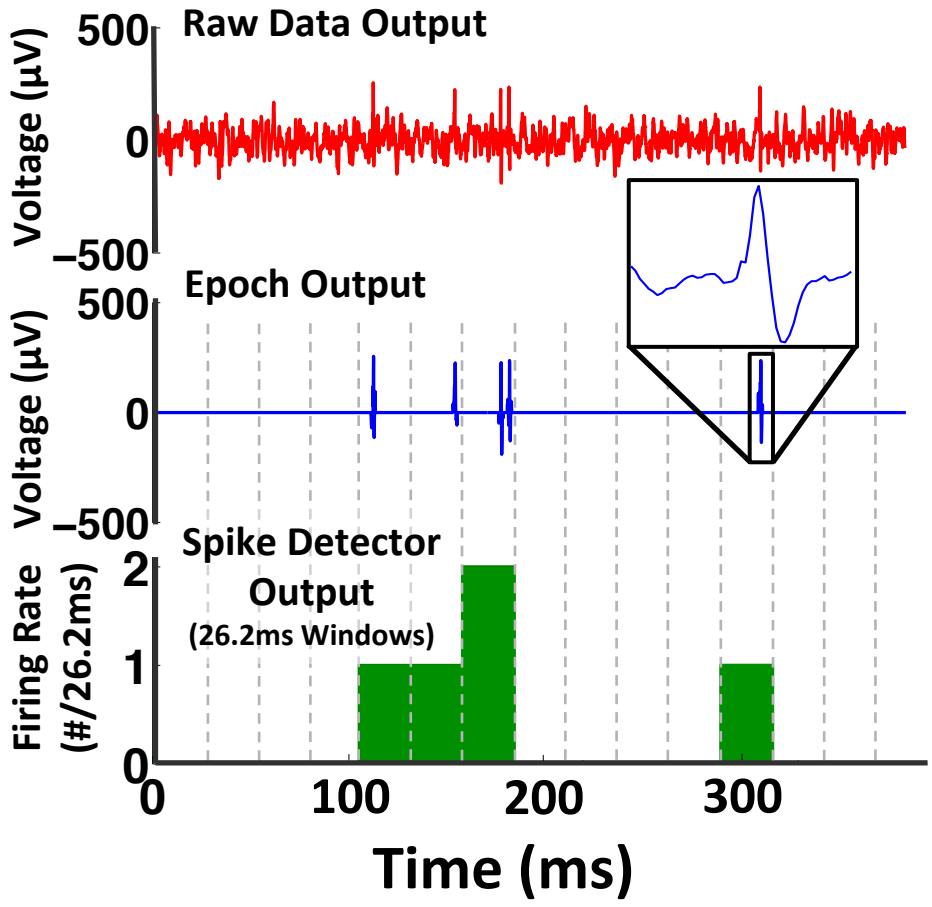


Figure 2.9: Uncompressed data, epochs, and firing rates from an *in vivo* recording.

As shown in table 2.1 the three different modes of operation are characterized for a specific dataset. The data rates vary from the maximum of 13.653 Mbps down to just 20kpbs to send the firing rates, a compression ratio of 700 times for the relevant data required for many algorithms that enable advancements in the field of BMI research. If the raw waveforms during those events are necessary, the epoch mode allows for a still more than 8 times reduction in the data rate, with the preservation of the data within those time windows to allow for more advanced post processing algorithms such as SVD. The numbers as show in table 2.1 are using values taken from the in vivo measurements and thus accurately reflect the system performance in proper operating conditions.

Data types	64 Channel Power	64 Channel Data Rate	Compression Ratio
Raw Stream	238 μW	13.653Mbps	1x
Epochs 50Hz/Channel	113.6μW	1.6384Mbps	8.3x
Firing Rates	77.63 μW	20kpbs	700x

Table 2.1: Power and data rates for different modes of operation

2.7 Learnings

By taking advantage of the fact that the signals used for this application are sparse in time, and often the algorithms that post-process the data do not actually look at the raw signal itself. Sending only the data that is actually required for these algorithms results in an up to 700 times compression of data rate, or in the system context would allow 700 times more channels for the same fixed data rate link. Now that there is less data to send the next steps are to work towards being able to transmit and communicate these bits as efficiently as possible. More measurement results from the combination of benchtop and in vivo setups can be found in table 2.2, specifically how the system implemented in [5] compares to other systems that have features such as the compression algorithm previously described, an integrated stimulator, and multi channel recording interfaces. The main rows of interest for the work discussed in this chapter are the compression/ DSP type and digital power rows, showing that it is the lowest power implementation while simulataneously having more modes of operation and data modes. A photo of the chip implemented in [5] is shown in fig. 2.10, showing the area of the digital compression and buffering in the middle of the chip surrounded by the amplifiers and simtulators on the edges.

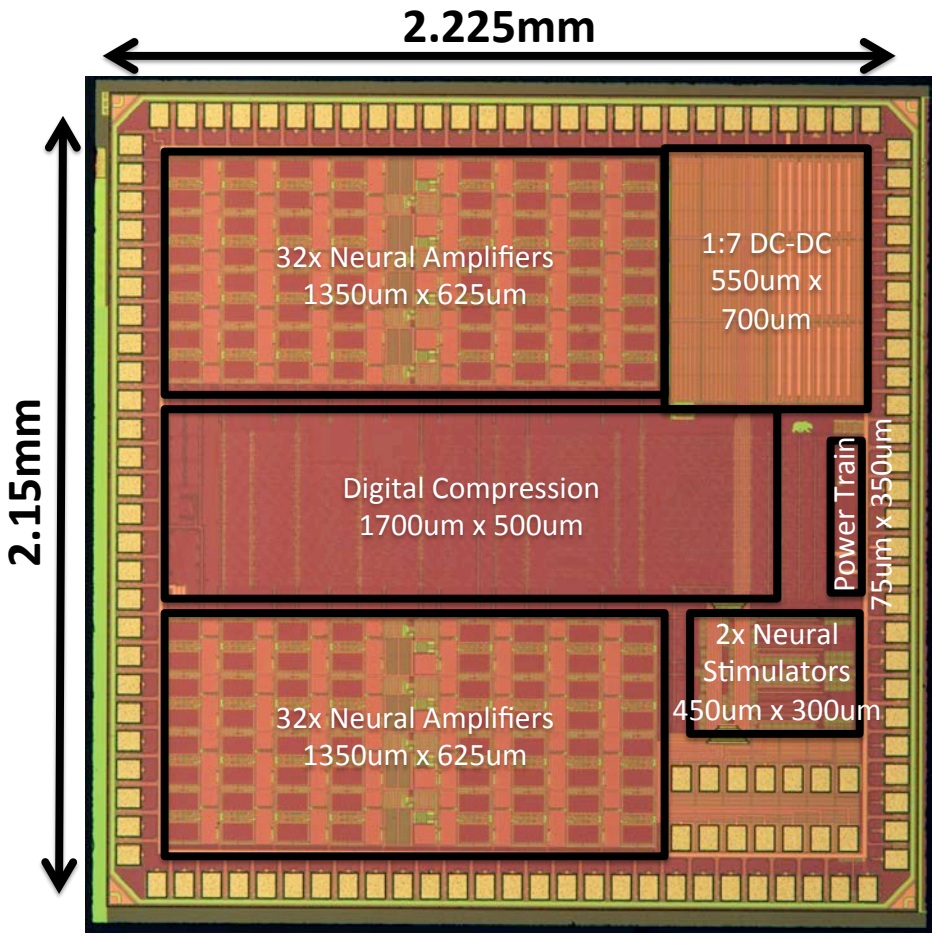


Figure 2.10: Die photo of the chip implemented in [5]

System Specs	This Work	[4]	[23]	[27]
Technology	65nm	0.35um	180nm	0.35um
VDD	1.0V, 0.8V (Ana, Dig)	1.5V	1.8V	5.0V
Off Chip Req?	None	1uF Capacitor	DC-DC	16-Ch. Recording IC
# Amp / Stim sites	64 / 8**	8 / 8	4 / 8	16 (Off-Chip) / 8
Amp & ADC Power / Area per site	1.81uW / $0.0258mm^2$	25.8 uW / $0.3122mm^2$	61.25uW / $0.354mm^2$	N/A
Gain / LP / HP	45-65dB / 10-1k / 3k-8k	51-65.6dB / 1-525Hz / 5-12kHz	54dB / 700hz / 6khz	N/A
Compression or DSP Type	Raw Data, Epoch, Firing Rate (any combination, per-ch.)	8 Spike Detector Outputs or 1 Ch. Raw	Log-DSP for LFP Energy, Ouptut Mode: 4Ch Raw	Spike Detections, Classification, PCA
Digital Power	1.21uW (FR) & 1.775uW (Epoch)	3.28uW	34.5uW	256.875uW
Area Per amp Ch.	$0.0105mm^2$	$0.0676mm^2$	$0.8mm^2*$	$0.191mm^2$

Table 2.2: Power and data rates for different modes of operation

The following chapters will discuss the additional blocks needed to make this into a fully wireless system, and how those same design technicques could be applied to a high speed serial link for a phased array system. There will also be more info about the respective projects that these links are targeting and why the approaches presented attempt to achieve those results.

Chapter 3

mmWave Burst Mode Wireless Link

3.1 Motivation

There are a number of wireless systems that require the entire system to be as low power as possible to extend battery life. This allows systems to be powered with energy scavenging techniques, or simply allows these systems to function for as long as possible between charging. The data rates required in these systems often are rather low or more frequently are variable depending on the way the system is configured or how it is going to be used in deployment. Instead of creating multiple designs for different applications it is desirable to create one design that is energy efficient at sending a wide array of data rates.

If we take an example of the system created in the earlier chapter about brain machine interfaces, the system has an interface that is collecting a large amount of data, and there is a clear need to be able to wireless communicate that information for both monitoring and data logging, with the additional benefit of enabling the use of more powerful computation elements. The chip presented in [5] can generate up to 16 Mbps if all channels are sending out all of their data modes at the same time, and there is a desire to enable a larger channel count than just 1 of those chips. So there is a need to design an energy efficient radio that can trasmit at least in the range of 50-100 Mbps in a small form factor and as low power as possible.

Looking at a few typical options used for sensor devices, bluetooth, zigbee and other simple protocols simply do not have a datarate high enough to be able to send this much data. Some of the higher order modulation wifi protocols would be able to achieve the data rates required, but the power required to implement those standards would not be feasible for a battery powered system small enough to enable the system of interest. With the additional constraint of desiring to have a communication system that is as efficient as possible across a range of data rates it is clear that the choice should tend towards some sort of a pulse based radio.

3.2 Potential Schemes

To efficiently send a scalable, efficient communication interface, there are two main methods: have a modulator that scales with the data rate, or have a fixed modulation rate and be able to duty cycle the radio over a large range. The first method would allow for a more conventional design that follows the tradeoffs that are well known in other applications, but most likely would result in a less efficient design from an energy perspective. The main reason behind this is that in many of these designs there are some components that consume a constant amount of power, and the system achieves a low energy per bit by simply increasing the data rate high enough that the static power consumption is a small contribution to the overall power. Since the data rates that are going to be targeted for this project are much lower the energy per bit of this static power consumption would make the radio very inefficient. Instead, the goal of this project is to only operate when it needs to, resulting in mostly dynamic power consumption that should scale with the data rate, not the frequency of operation. As discussed in the previous chapter, this goal would allow for a much larger range of data rates that you can efficiently operate over, especially if you are able to reduce the static power as much as possible.

Given that this wireless link is going to be heavily duty cycled, in order for the system to be as efficient as possible the static power consumption must be minimized. There are a few choices on how to move forward with some of the system level design considerations, mainly what frequency to operate at and what modulation scheme / type of modulation to use.

Since the data rates required for the projects that would benefit from this approach are relatively low by wireless standards, the first obvious choice would be to go with a low frequency and try to make that as efficient as possible. This would allow for a more straightforward design approach and less worry about more complicated electromagnetic effects and parasitics, and would be able to operate over a larger range due to lower transmission losses through the air. However, at these lower frequencies the available spectrum bandwith is much smaller, so radios are required to move to higher order modulation schemes in order to increase the data rates beyond a certain point. At higher frequencies there is a larger allcation of spectrum, allowing narrower pulses in time corresponding to larger bandwith used. Another benefit of the higher frequency operation is that the antennas would be smaller, allowing the overall system to fit in the desired form factor. In order to make the link be efficient across a wide range of data rates, the ideal approach would be to leverage other work that has previously been shown to have a higher data rate and be able to duty cycle the link in a way that when it is off it consumes as little leakage power as possible, and then by controlling the duty cycle you can efficiently change the data rate across a much wider range.

The modulation chosen for this project was intended to make the system easier to design, and to have to do as little calibration as possible. The main tradeoff here would be a higher order modulation scheme that can send multiple bits at a time, but would require

synchronizing clocks across the two wireless chips to within a low margin of error. Instead, one could go with a lower order modulation scheme to make the design as simple as possible, and design everything in such a way as to turn it off when not in use, and then rapidly turn everything on, send a single bit, and then turn everything off again. The scheme that would enable this properly would be on off keying (OOK) and thus the wireless system ends up looking rather similar at a high level to a high speed serial link, so there are many tricks that one could employ from that field for calibration and correct operation. Plus, the wide spectrum available at 60Ghz allows for relatively narrow pulses in time which correspond to large spectrum usage so at least for this version spectral masks were ignored. However, given the higher path loss at these frequencies one must either transmit more power or effectively transmit more power through the use of multiple elements together in a phased array system.

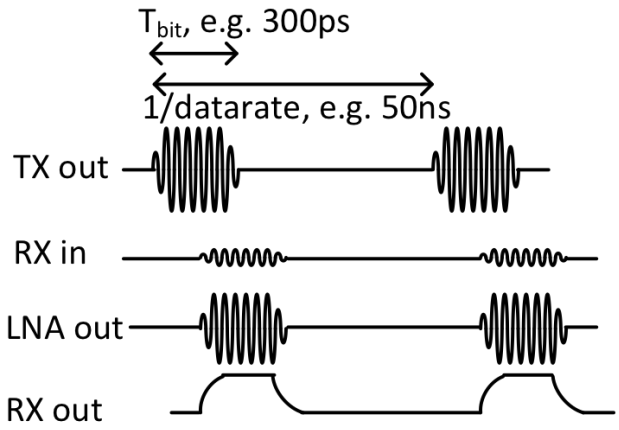


Figure 3.1: OOK Modulation Duty Cycle

A graphical representation of this scheme is shown in fig. 3.1. This example is showing the system sending a consecutive sequence of digital "1" bits, since when transmitting a 0 no energy is sent. This scheme is necessary for initial calibration, and there are limitations on how many consecutive 0's can be sent, but neither of these are enough of a limitation from the system perspective to warrant changing modulation schemes.

3.3 Architecture

The full system architecture including the phased array components can be seen in fig. 3.2. Each cell contains a transmitter and a receive front end, which in this case is just a tx/rx switch before going into the receive data chain. The system also contains a single RX data bath after and including the demodulator, since this is intended on being used as a single channel system as far as multiple input multiple output (MIMO) is concerned. There

is also an included clock and data recovery (CDR) and on chip clock generation, with the ability to override with an external clock for characterization and debugging.

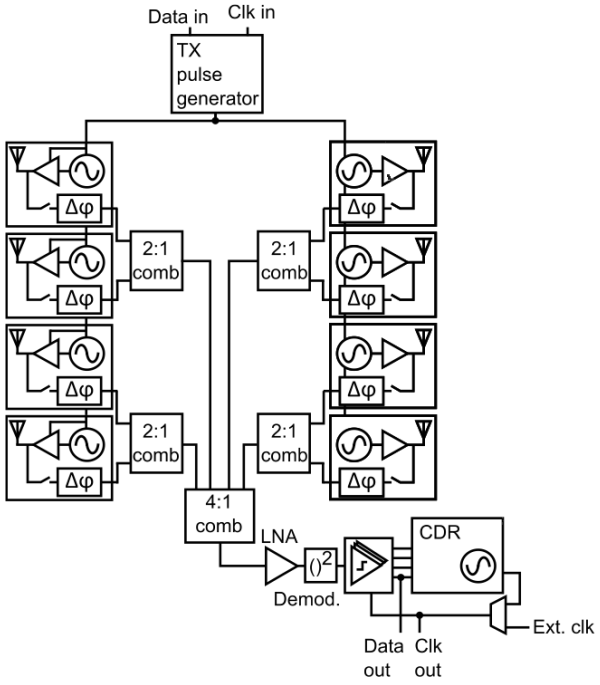


Figure 3.2: Phased Array System

The TX architecture becomes very simple given the chosen modulation scheme and system setup. Since the system is operating one bit at a time, and the TX simply needs to send a "1" or nothing at all the architecture is rather straightforward. If the system were not a phased array, all that would be required is a pulse generator and a fast start / stop oscillator and buffer to drive the antenna. With multiple elements the only addition is a phase adjustment between the different elements, implemented as a per element delay control. Because of process variation between different oscillators there is also a per element varactor control for frequency control of the oscillator. The architecture of the oscillator is an adaptation from [14]. A simplified schematic version of the transmitter, which is comprised of the oscillator and amplifier is show in fig. 3.3.

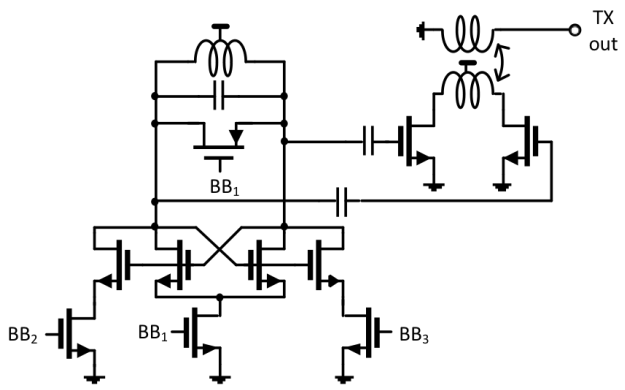


Figure 3.3: TX architecture

In order to implement the full circuitry the bias points on the amplifier inputs need to be designed to settle before the transmitter is turned on. In addition, the architecture of the oscillator is such that one side, in this case the right, needs to be turned on before the other two devices, imposing an initial condition across the LC tank and therefore enabling it to startup quickly. The necessary delay cells as well as the power up bias circuitry are shown in the more detailed diagram in fig. 3.4. These two elements combined ensure that no extra energy is sent to the antenna when the system is not meant to transmit anything.

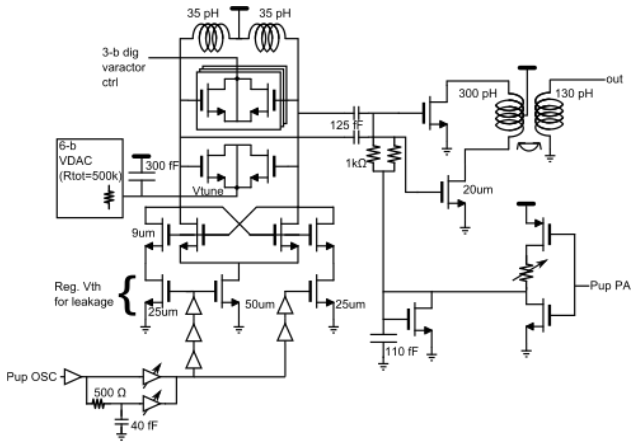


Figure 3.4: More Detailed TX Diagram

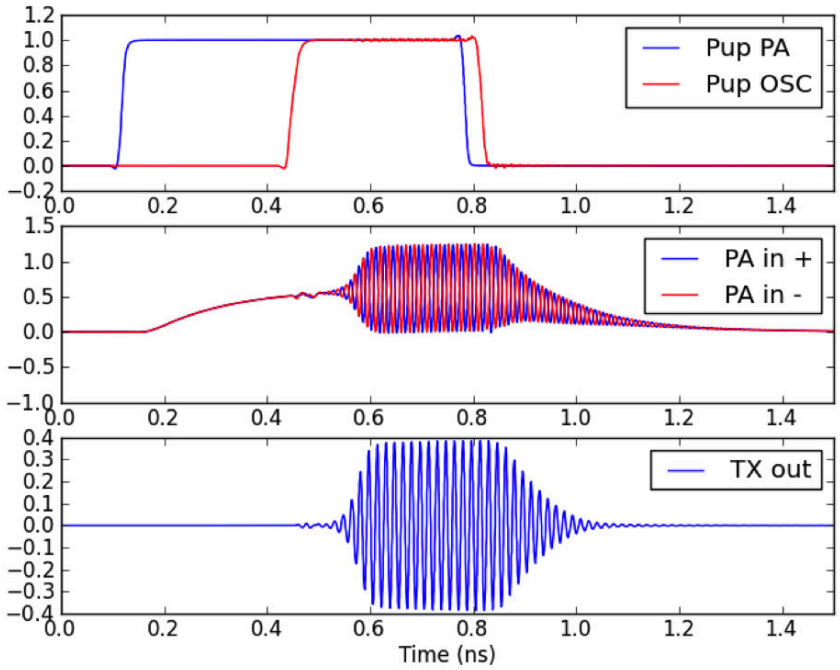


Figure 3.5: TX Transient Results

The receiver consists of a TX/RX switch, a front end phase shifter and combiner to a single signal chain, as shown in fig. 3.6. The only array element processing here is the ability to shift the phase of each of the incoming streams independently. This is mainly due to the OOK nature of the system, where as long as the signals are not adding destructively you can still obtain a benefit from adding more elements.

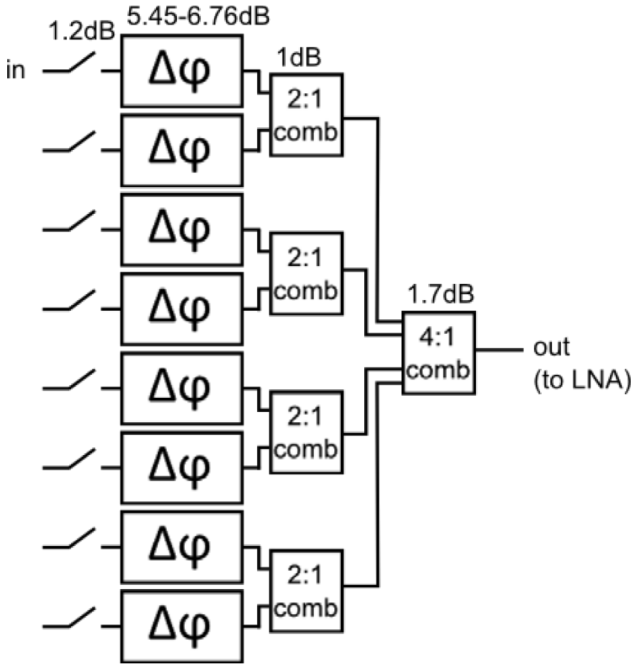


Figure 3.6: Phase Shifter and Summer

After amplification as shown in fig. 3.7, the signal passes to a self mixing demodulator, as shown in fig. 3.8. This architecture was chosen to limit the amount of biasing needed to help reduce the total amount of standby power required, and also achieve the performance specs required without adding in undue complexity.

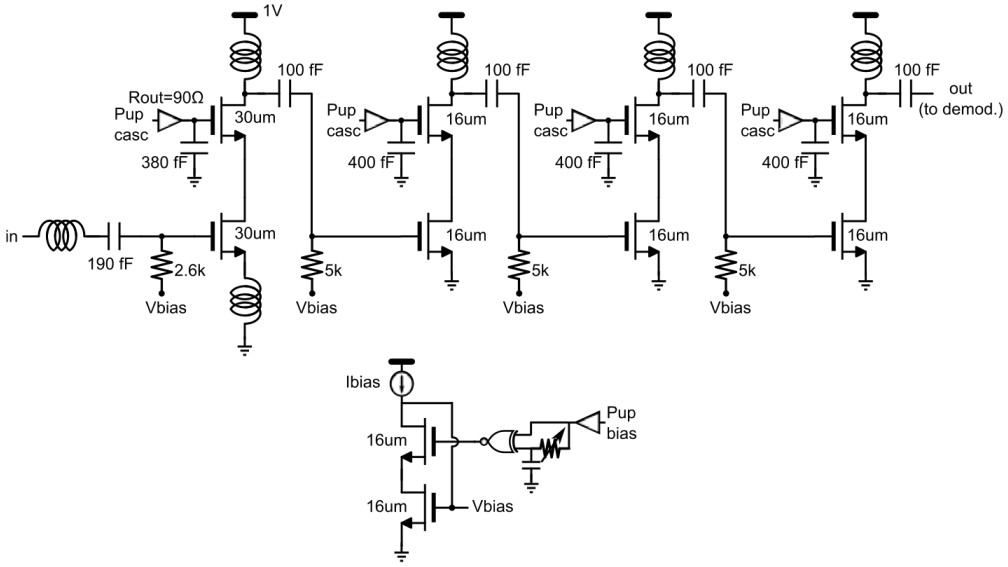


Figure 3.7: LNA Schematic

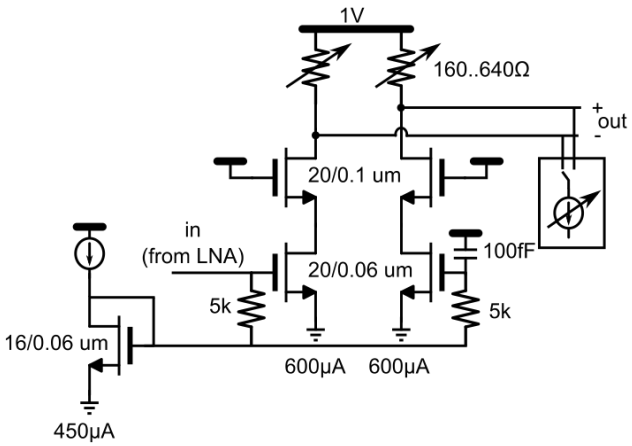


Figure 3.8: Demodulator structure

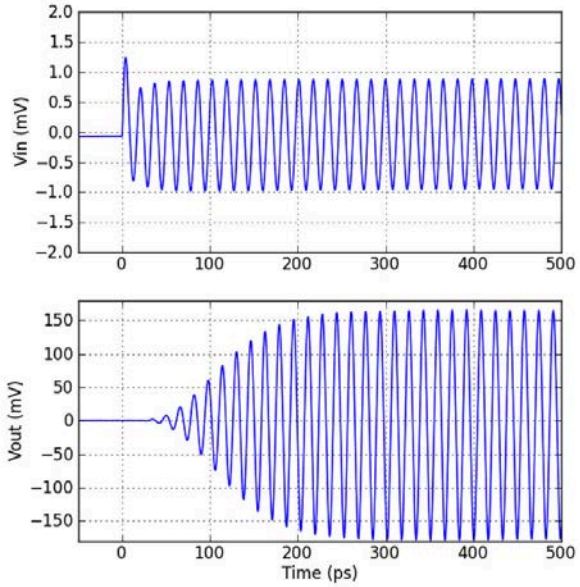


Figure 3.9: LNA Step Response

Simulated step response results are shown in fig. 3.9, showing that in simulation the frontend of the receiver can startup in under 200ps.

3.4 CDR and Calibration Loops

A block diagram of the full receive chain is shown in fig. 3.10, using the blocks that were discussed in the previous section. The focus of what I did for this project is mainly on the baseband and calibration loops required to get the link to operate properly, so this diagram can be useful as a starting point to discuss the required blocks in terms of both circuits and digital loops to enable the desired behavior.

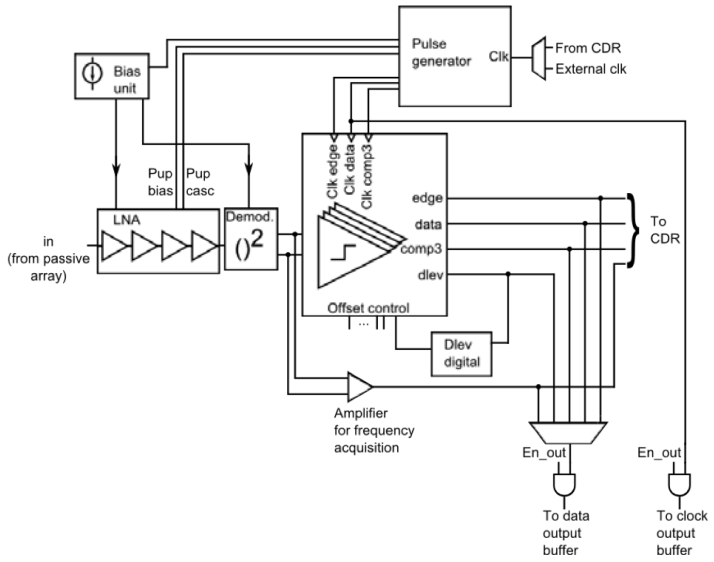


Figure 3.10: RX Array System

Even if we already know what data rate we would like to be set to, since this is a wireless system and inherently is a communication interface between different chips that may not be connected to each other, this design requires the ability to recover a clock from the incoming data stream. Because of the modulation scheme chosen, nothing is sent when the data bit is a "0" and therefore there is a maximum consecutive run length of "0"s before the lock is no longer guaranteed from this recovery scheme. The pattern is long enough that for random enough data this constraint is not a burden, but if this were to be an issue a simple coding overhead could be added to guarantee some balance of data bits. For simplicity, the system will be explained in the case that the transmitter is sending solely "1" data bits throughout the calibration sequence. In the case when data is being used, the updates are only valid when the data bit is a "1", otherwise the update is not used. This imposes a limit on the amount of consequitive "0's" as well as the overall number of them transmitted during operation, but simulations show that in the worst case a scrambling pattern can be used to guarantee the loop will still lock properly.

When the system is initially powered on, there is no guarantee of any relationship between the internal clocking of the receiver and the incoming data, so the scheme that will work

in steady state is basically useless. For this initial acquisition a continuous time path is implemented for the purpose of acquiring a close enough initial phase to be within the locking range of the subsequent loops. In the worst case, the rising edge of the on chip clock and the incoming data pulse have completely no overlap, and all of the samplers will read the same value. In this case, the loop assumes that the clock is late, and continues in that direction until the pulses overlap at least slightly. Once this has occurred, the loop will have a setup allowing it to work properly, trying to ensure that with three samplers it can successfully sample the edges and the middle of the incoming data that is turned into a pulse by the demodulator.

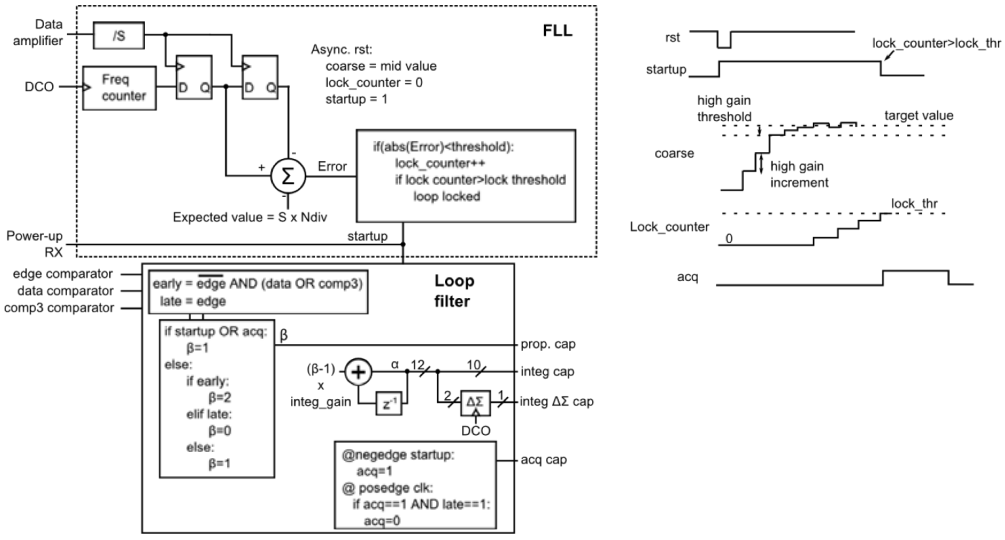


Figure 3.11: CDR Acquisition Diagram

As show in fig. 3.11 the continuous data stream is sent to a frequency locked loop (FLL) that determines the necessary settings to ensure that the frequency of the incoming recovered clock matches the clock the the DCO generates. Once the frequency is locked, the phase needs to be determined as well. A modified version of the steady state locking mechanism is used to adjust the phase of the oscillator to match the incoming data stream. In this scheme the internal state of the loop filter is initially set to its lowest value, and upon consecutively seeing that the comparators do not see the data pulse the counter is increased until that is no longer the case. In order to improve the locking time of this linear search, a larger gain setting is applied initially to effectively perform a more coarse search to find the pulse initially, and once that is found a lower gain setting is employed to improve the resolution of the loop. Once the loop has found the pulse, the continuous time path can be disabled to save power, and the receiver can enter burst mode operation.

In burst mode operation of the receiver, a scheme similar to a 2x oversampled CDR for a high speed SERDES is implemented. Three comparators are used to sense the timing of what the receiver thinks is the middle of the pulse relative to the edges of the pulse and can

adjust its phase and frequency to continuously track the pulse. In the case that the initial loop has already locked, the middle of the three comparators in time will be somewhere within the pulse, and the other two comparators will be offset in time by some amount. By looking at which one of these two comparators is a digital "1" the system can know whether it is "early" or "late" and adjust its phase accordingly. Simultaneously, a "dlev" loop tracks the analog level of the incoming data in the middle of the pulse to try and figure out an estimate of the signal level to enable beamforming and other debugging information. This is implemented as comparing the analog value to a DAC and looking at the sign, which is used for a sign-sign LMS loop.

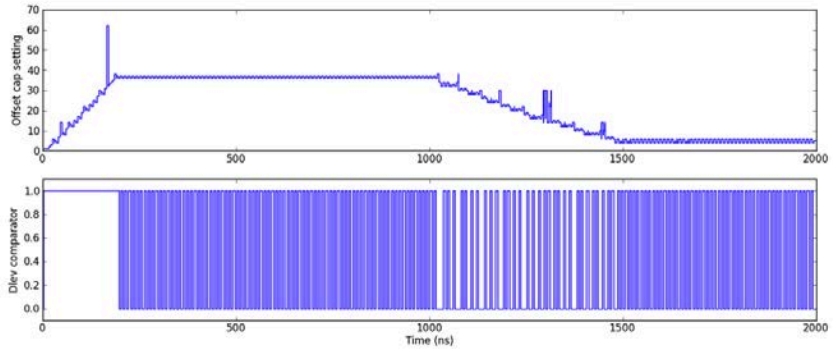


Figure 3.12: Dlev Loop Simulation Results

System level simulation results of the dlev loop converging are shown in fig. 3.12. From this simulation the loop is able to receive the correct data as well as track the data level of what the analog level of a "1" is, with the simulation chaing the amplitude of the incoming waveform at 1000ns into the simulation to show that the loop is able to track those changes.

Datarate	Tbit, min	Min DC Power	Energy per bit
100 kb/s	7.1 ns	23.3 µW	233 pJ
1 Mb/s	1.5 ns	66 µW	66 pJ
10 Mb/s	330 ps	300 µW	30 pJ
100 Mb/s	70 ps	2.2 mW	22 pJ

Table 3.1: Power and data rates for different modes of operation

3.5 Learnings

By simplifying the modulation scheme and the system as a whole, the system allows the use of available spectrum at 60Ghz to efficiently create a wireless link that has a low energy per bit across a wide range of data rates. By utilizing the large amount of available spectrum

to design a radio that trades off spectral efficiency in terms of making narrow pulses in time which correspond to a wide amount of bandwidth in the frequency domain, one is able to design a radio that is able to rapidly transition from its low leakage off state to operation. This system level setup allows for a large scaling of data rates where the link itself is energy efficient, and would even function beyond that range at a hit to efficiency.

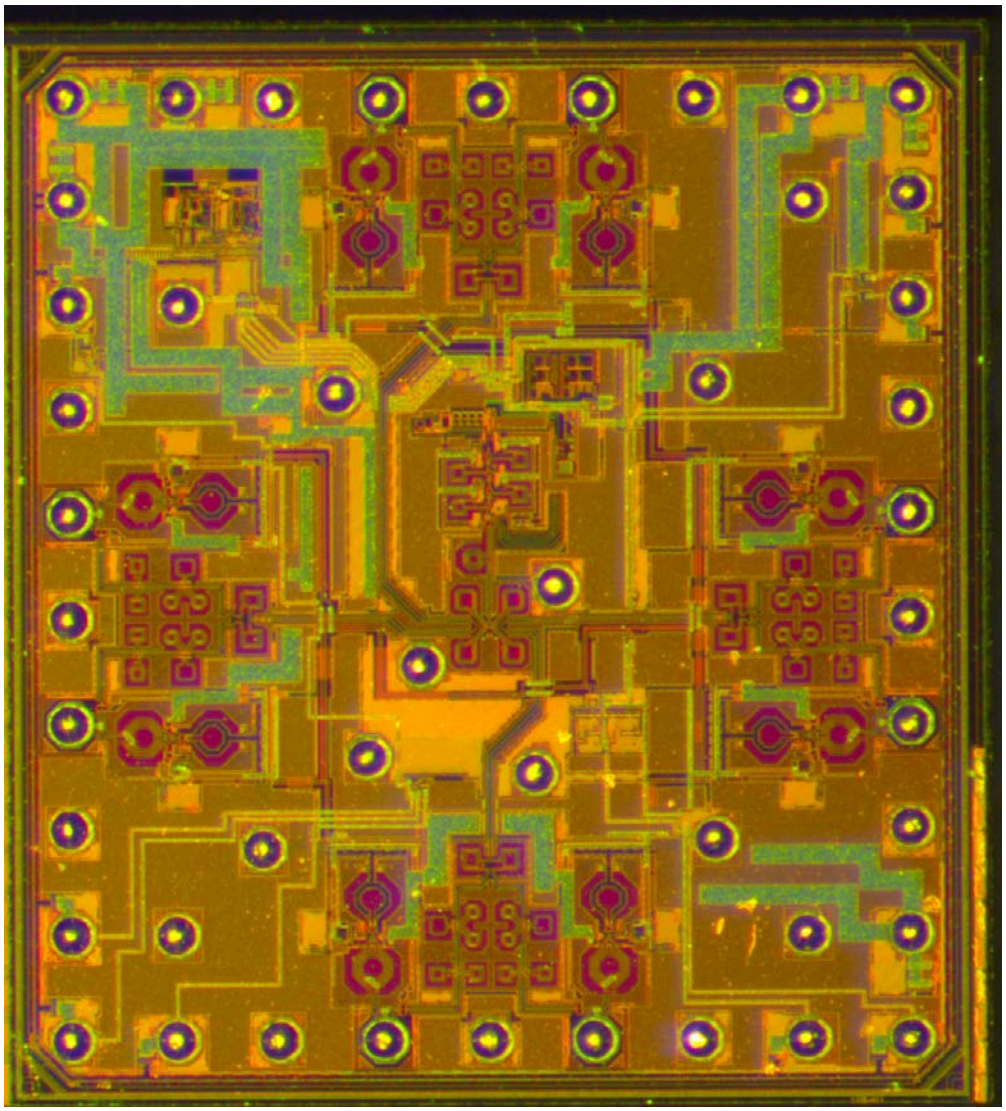


Figure 3.13: Die Photo

A die photo can be seen in fig. 3.13, however unfortunately due to an implementation issue with either the pad drivers, the on board buffers, or their interaction the interface that was supposed to have several hundred Mhz of bandwith was measured to only have 2 Mhz.

Given that the chip had no other testing interface it was not possible to measure the results of the implemented circuits.

Chapter 4

Burst Mode Serial Link

4.1 Motivation

Building upon the same concepts of the burst mode wireless link in the previous chapter, one could apply those same design principles for desiginging a serial link, if there is an application where such a design makes sense for the system. The project that will be covered in this chapter is titled eWallpaper, with the goal of buildling a flexible array of digital efficient radios based on a . The idea of this project is that you want to use a single common module chip that is replicated multiple times to create a large array of radios, and by building each of the radios to be as simple as possible overall the system can be more efficient than a single radio. In order to achieve this, a lot of care has to be put into not only building the radio components themselves to be as efficient as possible, but more relevant to this part of the work is that everything else needs to be as low overhead as possible.

In order for the system itself to function properly as a phased array all of the chips in the array need to be synchronized and fed the same data, so there needs to be a serial link to communicate this. The additional constraint of building a system that is meant to be assembled on a flexible substrate means there are limited metal layers for routing, so links would only be between neighboring chips and a large array would incur the number of chips in a column/row as the minimum number of hops, or that divided by 2 if there are connections on top and bottom (or left and right). A picture of what this looks like is shown in fig. 4.1, where the blue squares represent different individual chips and the black lines are the connections between those chips as well as the drawn antennae.

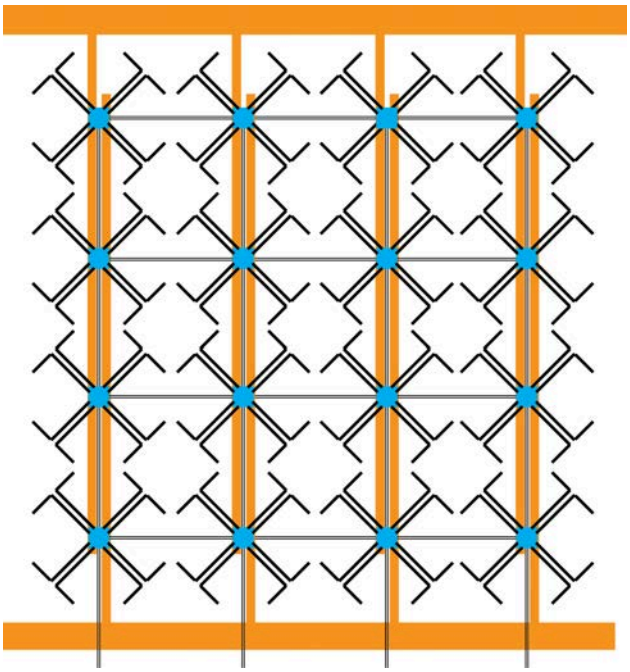


Figure 4.1: eWallpaper System Diagram

More information about the details of this approach can be found in both [21] and [22], but the main focus here is on the links within the system, so specifically looking at equation (4) from [21] we see that the total power of the system is dictated by

$$P_{tot} = \frac{EIRP}{M\eta} e^{(\frac{k\lambda}{2}\sqrt{\frac{M}{N}})} + MP_{tx} + NP_{chip} \tag{4.1}$$

The first term in this equation is specifically talking about the RF performance and the routing length of the antennae, which for this discussion only impacts how far the serial link needs to communicate across. The second term covers the overhead power per module, where M is the number of transmitters in the RF phased array system and the overhead power is anything that does not contribute to the RF transmitter outputting radiated power. This means that something such as the clock generation for the TX itself, or any bias circuitry would count towards that, as well as the power required to synchronize the data between the different chips in the array. The third term is covering overhead power that is per chip, which would be items such as global clock generation and distribution, chip level processing that needs to occur on the data, or DC-DC generation inefficiencies. Specifically any power for a serial link to tranmit the data between different chips in the array is included in that term, so the total power of the array will increase the more energy the serial link consumes. There are second order effects of that overhead power increasing as the latency of the serial link increases due to a larger amount of buffering and range of synchronization that ends up being introduced.

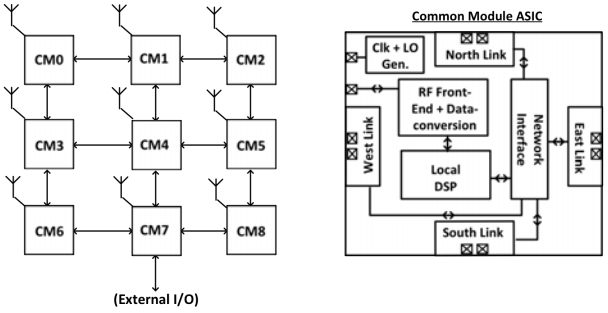


Figure 4.2: Common Module System Diagram

As seen from [1] a typical server workload is such that the utilization of the memory interface varies over time, with typical numbers suggesting that on average between 10 and 50 percent of the maximum bandwidth is used aggregated over time. This means that even in the most general purpose application with conventional serial links they woudl be consuming between 2x and 10x of the necessary power simply sending nothing to keep the serial links synchronized. Conventional serial links that have power states have typical switching times in the order of microseconds, meaning that the minimum time required to switch power states makes it prohibitive to save power except in the cases of long term idle workloads, which are not that frequent. So while the burst mode approach is beneficial for the specific application given, it can also be more broadly applied and thus it is not surprising that other people have tried this before, as seen from [2, 30, 17]. After going through the specific blocks and how they need to change to enable rapid turn on and off, some system performance numbers will be covered and compared to other works.

4.2 System Architecture

To explain the need for different data rates, one needs to understand the difference between centralized and distributed beamforming. In an effort to simplify the design of the components on the chip itself you could imagine putting all the necessary computation to perform beamforming and channel estimation on a more powerful external device, like an FPGA or a full blown computer sitting on the edge of the array. In this case all of the data needs to flow to this external device where the centralized computation takes place. In the RF system transmit context this would mean that you precompute the exact values that each antenna would ultimately send on this external device, so you do not need to distribute beamforming coefficients at all, and the transmitter would simply have to send the data that it is given. A graphical example of this is show in fig. 4.3. As should be clear from even this picture is that as the number of chips in the system increases the data rate for the edge interface increases linearly, so scaling to large arrays becomes very challenging. Simultaneously, the link shown in the top left in this configuration sees a quarter of the data

rate of the maximum, and that fraction only gets smaller as more elements are added. To create something that would be efficient across this wide of a range of data rates is one of the motivations behind the burst mode type architecture.

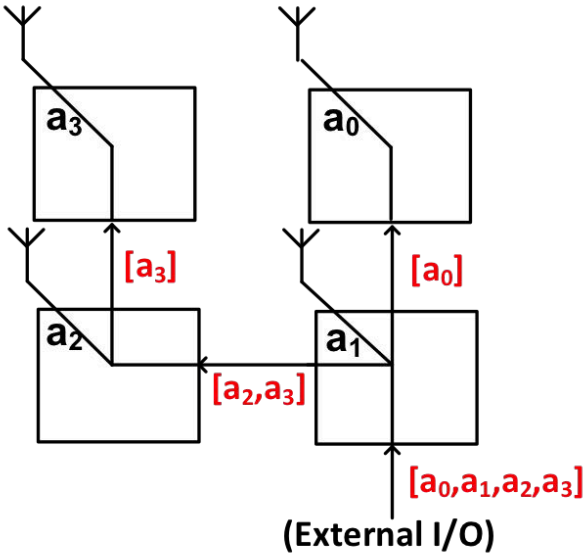


Figure 4.3: Centralized Beamforming

In the case of sending out all of the data to all of the individual chips and doing the computation there (or distributed beamforming, as seen in fig. 4.4) all of the links in the system would see the same data rate, in this picture the size of d1 and d2 times the rate they need to be updated. But upon closer inspection this represents 2 simultaneous streams of information that are attempting to be sent at the same time, each with their own beamforming coefficients (all of the α's in the figure). This means that depending on the configuration of the system itself the data rate of all of the links will need to change, again with a large degree of scaling between the minimum and maximum required. In this example the range of scaling for distributed beamforming would need to be the ratio of the maximum supported continuous beams to the minimum number.

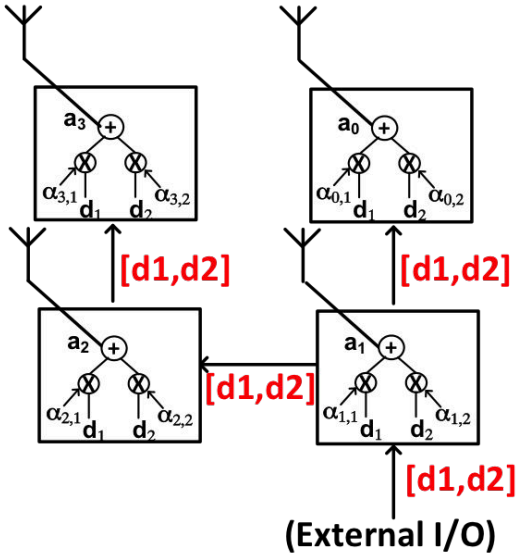


Figure 4.4: Distributed Beamforming

4.3 Circuit Architecture

At a high level, all high speed serial links can be broken down into a small set of sub components. There is a clocking system, some sort of equalization and digitization or simply converting the analog signals to bits, and an interface to the lower speed digital domain, typically a serializer on the transmitter and a deserializer on the receiver end. There are versions of the typical approaches for designing these blocks for conventional serial links that lend themselves better to a burst mode system, but ultimately they will each have their own design tradeoffs that lead to a different architecture for the entire system to start and stop quickly while burning as little power as possible in the off state. In prior work from this research group the focus has been on receive side equalization to not limit the maximum output swing on the transmitter, as well as trying to make the system as energy efficient as possible. As an added benefit this removes the requirement for a backchannel to adapt the coefficients. This thankfully is also an architecture that is inherently clocked with limited bias and other blocks that are not ideal for burst mode operation, so the receiver itself can be used with limited modifications. The rest of the blocks are going to be discussed in more detail as to how the design tradeoffs are typically captured and how that process changes for a burst mode system. These key blocks are shown in detail in fig. 4.5 with the final versions.

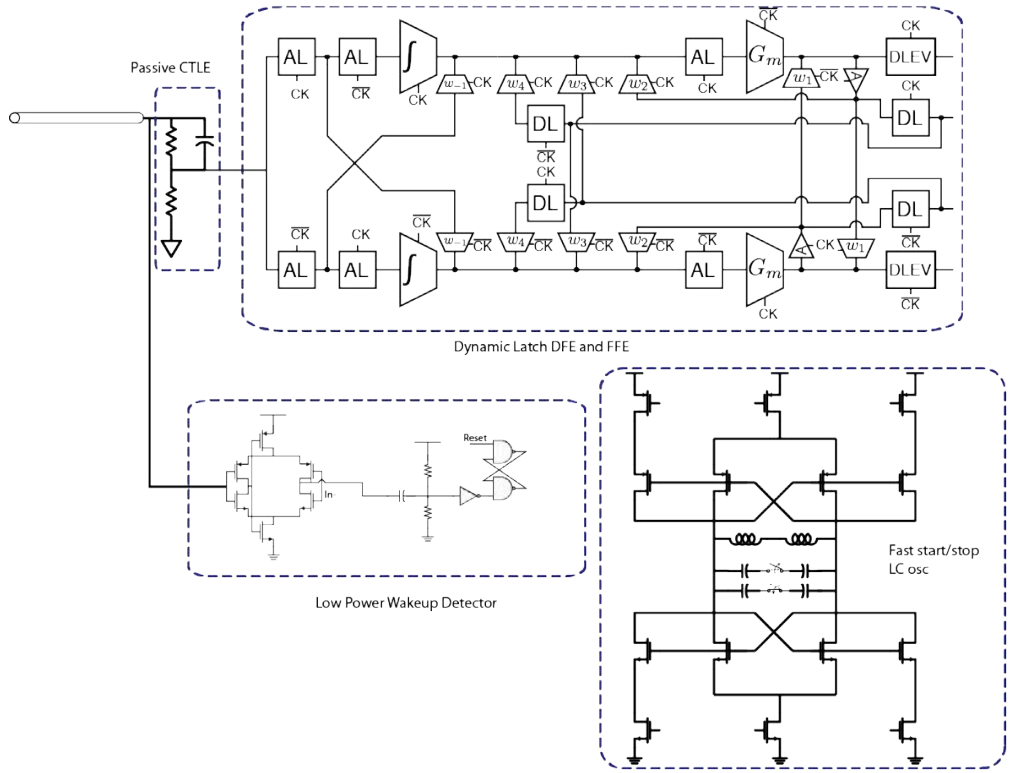


Figure 4.5: Link System Overview

Equalizer

Conventional equalization systems use a continuous time linear equalizer (CTLE) in conjunction with some number of taps of feedforward equalization (FFE) and decision feedback equalization (DFE). Conventionally the coefficients for these systems are high precision current DACs that would impose a large constraint on the startup / cool down time without a considerable amount of effort for that purpose. Instead, we utilize a passive CTLE that consumes 0 active power for its operation, and a dynamic latch based FFE and DFE, with bias DACs for the clock nodes as well as enable switches to turn off the latches when they are not in operation. A schematic of the dynamic latch can be seen in fig. 4.6.

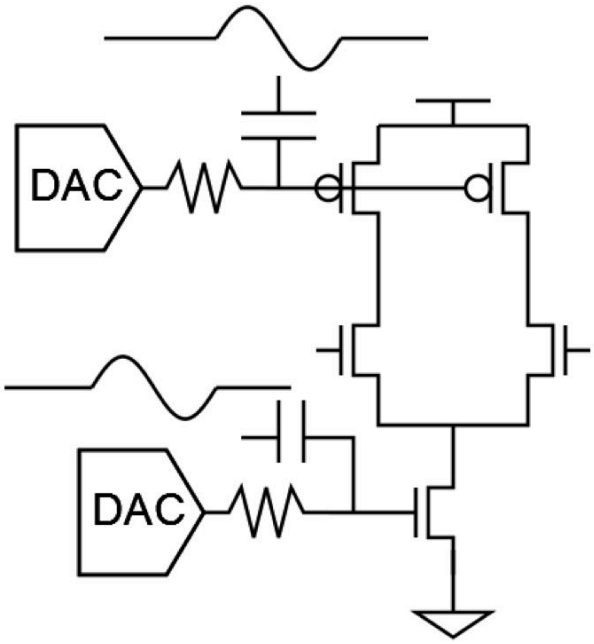


Figure 4.6: Dynamic Latch Operation

Since the intention of the latch is for the tail nodes to settle during every cycle, the startup constraint becomes much less of an issue, and if you bias the latch properly the AC clock pulses will turn on the transistors, and only leakage current will be flowing when the intent is for the latch to be in its off state. If the leakage current for the process is too large, you can add a series enable switch to the clocked current source device with little to no impact on the startup time of the devices. The only negative impact of this is the limited headroom that you are now imposing on the current source device since the timing on the enable signal is required for other parts of the system already. More on this specific info later, but in order to operate the system properly there are several calibration loops required.

Clocking

For the clocking system, there are basically two types of clocking options: a high quality factor LC tank based approach, or a ring oscillator based approach. An example of a LC tank based approahce can be seen in fig. 4.7 as seen from [8].

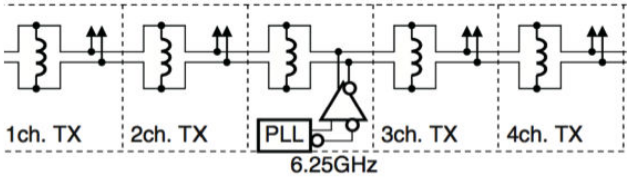


Figure 4.7: Example of resonant clocking network

A ring oscillator based approach can be seen in fig. 4.8 taken from [19].

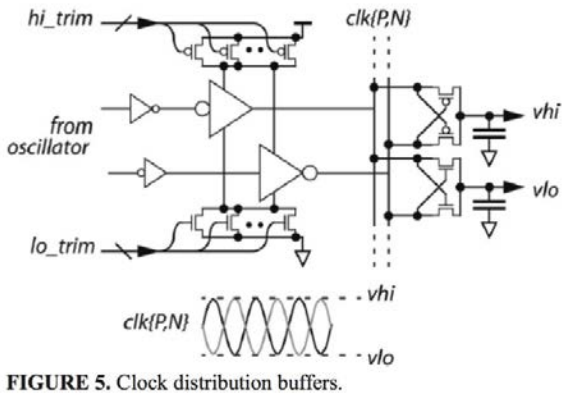


Figure 4.8: Example of ring oscillator clocking

Conventionally, the startup time of the ring oscillator is much faster than the high-q tank because the bandwidth is much higher and more of the noise to start the oscillator falls in band. However, we can break this tradeoff and achieve a high-q oscillator that has lower phase noise, better energy efficiency and a fast startup time. In order to do this, extra devices are added to allow a known initial condition to be enforced across the LC tank. Under this condition, the startup time of the oscillator is reduced the higher the quality factor of the tank. This seems opposite of conventional wisdom, and that is due to the fact that people are usually talking about startup time solely due to noise. Adding a kick to the tank changes this analysis, and also guarantees that the phase of the oscillator is constant on all future bursts, relative to the enable signal that is starting the oscillator. More information on this phenomena can be found in the work by a former student, Linkgai Kong [14]. A schematic of the complimentary oscillator can be seen in fig. 4.9, the complimentary version being chose to limit the swing to be rail to rail. In building this for a full system there is a desire for the entire link to operate on a single supply which takes out the ability to change the supply voltage to change the swing of the oscillator. The complimentary version also ensures a symmetric waveform, since the output of the oscillator is used directly we actually care

only about the two single ended waveforms and not the differential waveform. Post layout extracted simulation results of the oscillator can be seen in fig. 4.10, showing that it is able to startup in under 600ps to full swing driving the entire loading of the clocking system.

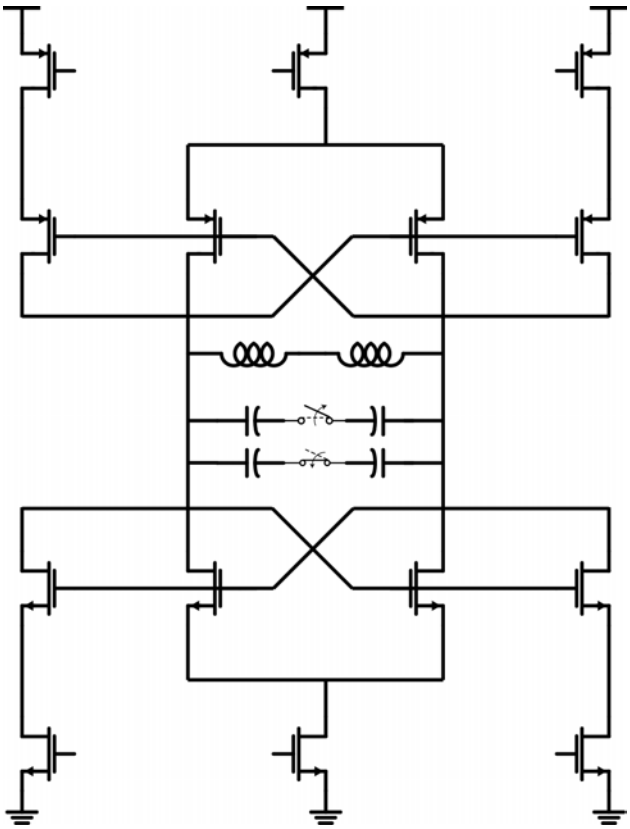


Figure 4.9: Oscillator Schematic

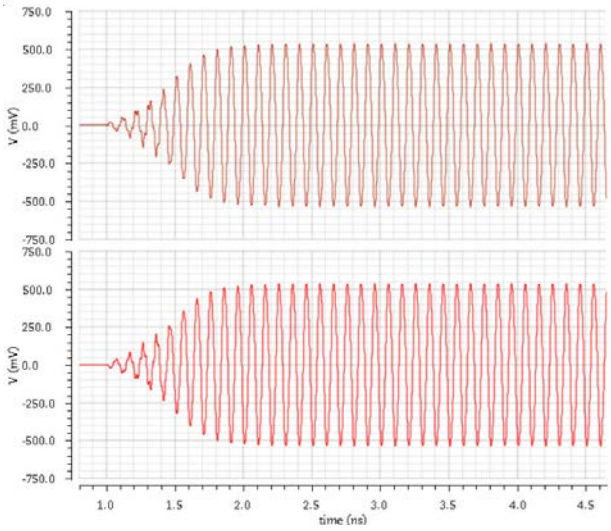


Figure 4.10: Oscillator Simulation Results

Wakeup

Since there is nothing specifically starting the link in a synchronous way across the different chips aside from a potentially noisy digital clock with an unknown phase relationship there needs to be an element in the link itself on the receiver that knows when it should wake up. Because of the dynamic latch operation of the receiver itself being clocked, using that to figure out when there is a clock would be challenging. Instead, a low static power wakeup detector is used to determine when there is a detectable differential signal on the line, and uses that with a calibration loop to compensate for delay variations to wake up the receiver as well as the receive oscillator. The structure of the wakeup detector can be seen in fig. 4.11, which is a self biased differential to single ended amplifier with high gain. This allows for a high gain amplifier that when in nominal conditions it is mostly railed one way or the other, so the current consumption is mainly dominated by leakage. The gain of this stage is high enough that subsequent processing can be simplified and almost setup as fully digital circuits.

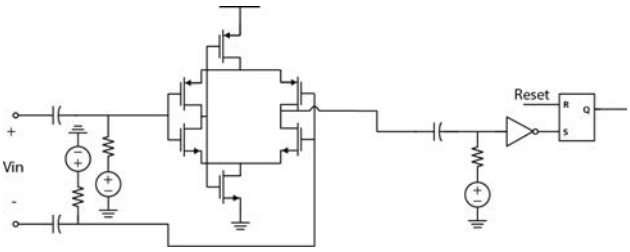


Figure 4.11: Wakeup Detector

The output of the amplifier is AC-coupled to the input of an inverter, which is biased in such a way that it is railed one way, and the incoming signal is more than strong enough to overcome this threshold, triggering the set reset flop so that its output will go high. The intent of this is that it will get set at the start of the pulse, and a different signal will be used to activate the reset port of the set reset flop. You cannot use the same mechanism as is to trigger this reset because in the off state because in the transition from the off state to the on state goes from a differential 0 to a differential positive one, which is then followed by a series of data bits that are either differential positive ones or differential negative ones. Then in transitioning to the off state it will either be at this differential positive one or differential negative ones to the differential zero, which is a transition that happens as a part of all of the transitions in the data themselves all the time. Instead of trying to figure out how to do this and rely on the edge detection here and figuring out that the pulse has been gone for some thresholded amount of time, higher level information is used to figure out when the "packet" is done with. For this version this is done as a counter on a divided clock to get an integer number of words in the packet length, which could be programmable based on a header in the packet itself. For simplicity this version uses fixed size packets, but one could easily extend this scheme to change that and be able to tradeoff packet size and idle time for the desired data rate and throughput of the given application.

Figure 4.12: Packet Timing

The timing on the first bit of the wakeup is the most important, since the closer you are to being at the correct time the less accurate the rest of your clock and data recovery has to be, especially for short packet sizes. The packet timing looks as shown in fig. 4.12 and introduces a few positive ones to make the wakeup detector have to deal with a lower bandwidth signal, followed by a 010 pattern that will enable the scheme for determining what is the timing of the wakeup relative to the incoming bitstream. This is then followed by the payload of information, with a footer that is shown in this image to be the same as the header but does not necessarily have to be.

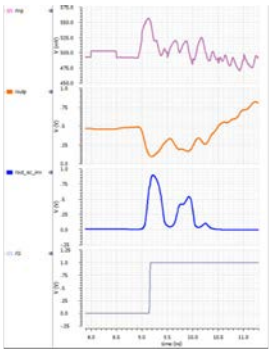


Figure 4.13: Wakeup Detector Simulation Waveforms

As shown in fig. 4.13 simulation results show that the entire wakeup detector has a startup delay of 200 ps.

Transmitter

Due to time constraints a TX was not included in the tapeout, but a similar analysis was done for that key block. An architecture that relies on precision current DACs would again not be a suitable candidate for turning off and on quickly, so a CML transmitter is not an ideal candidate. Conventional voltage mode transmitters use transistors to generate the 50 Ω impedance required to match the lines, and usually do so with a regulator for the cells driving the output devices. This implies that you would need a regulator that operated continuously and would be an overhead in terms of power and also most likely startup time. Both of these undesired architectures can be seen in fig. 4.14.

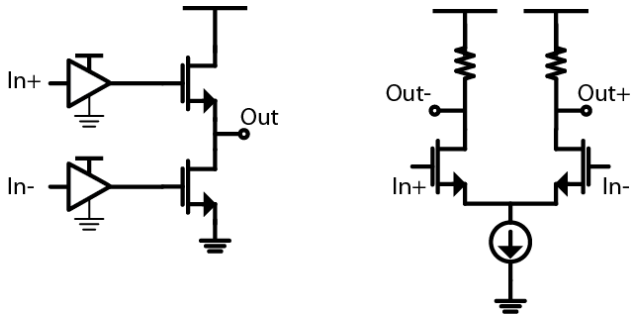


Figure 4.14: Conventional TX Architectures

In addition to requiring static power or having a longer startup time, the CML architecture has the added constraint of making it difficult to introduce a necessary 3rd state that is required for the wakeup system discussed previously. The normal two phases on a continuous time transmitter are shown in fig. 4.15, which is usually not an issue for the vast

majority of serial link systems. However, to easily detect that the system is in the off state a third phase is introduce, as shown in fig. 4.16. The 3 phases are differential positive 1 (+1), differential minus 1 (-1) and the newly introduced differential 0. You could create this 3rd state by powering off a transmitter, however this usually would mean that turning off and on quickly would require a large amount of power due to settling constraints.

Figure 4.15: TX Differential Signal

Figure 4.16: TX Differential Signal - Additional Phase

The approach that satisfies all of these requirements would be to use a switched capacitor based transmitter as demonstrated first by [20]. This design allows a capacitor to be charged during one phase of the clock, and on the other phase that charge is added constructively or destructively to the output, depending on the data bit that is being supplied. This charging phase is demonstrated by the switches in fig. 4.17, allowing the supply and ground to be connected across the capacitor which charges it. During this phase the switches are configured such that the switches connecting the capacitor to the supply and ground are closed, creating a pathway for the capacitor to be charged.

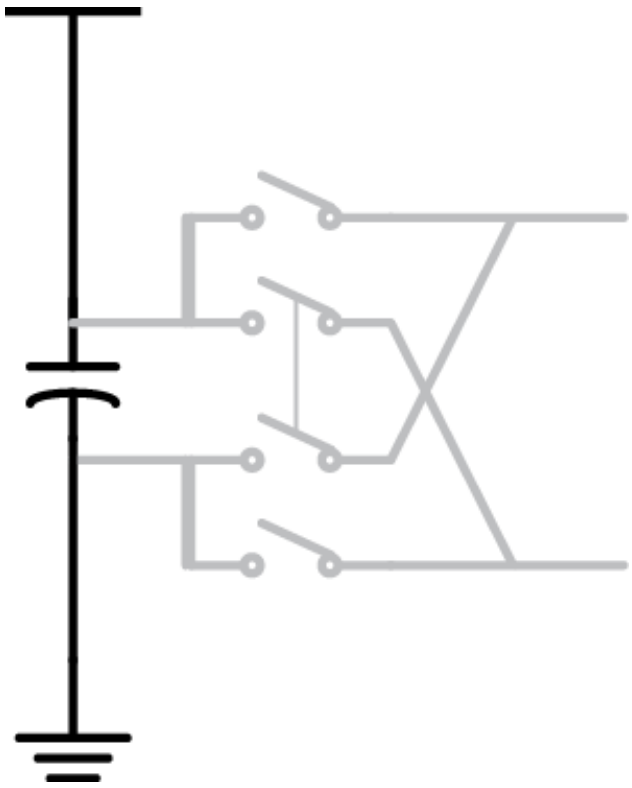


Figure 4.17: TX Charge

Inherently it should make sense that during the other phase, this charge would somehow be connected to the output, and the nature of that connection would be dependent on what the value of the bit that is being transmitted is. If the data is a digital 1 the connection would be completed as in fig. 4.18, connecting the positive side of the now charged capacitor to the positive differential output, and the negative side of the capacitor to the negative differential output.

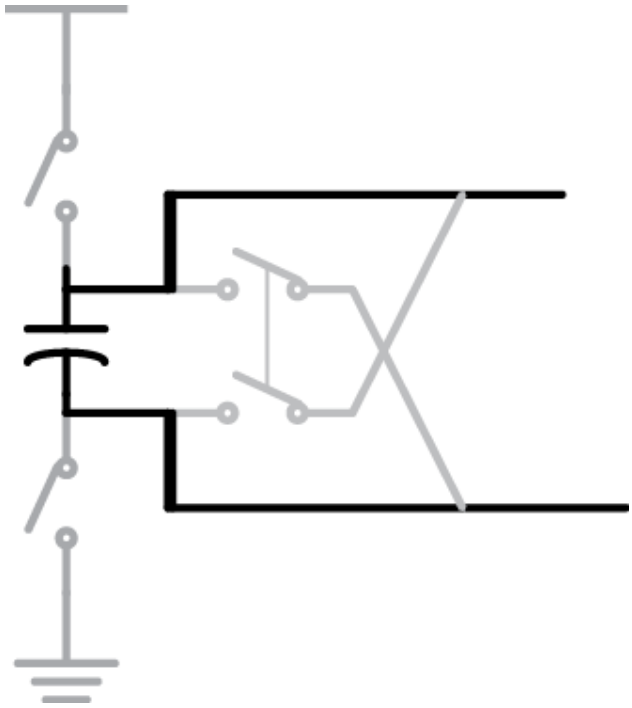


Figure 4.18: TX Plus One

If the data is a digital 0 the polarity of the charge with respect to the output would be reversed as shown in fig. 4.19, using the other set of switches that would connect the positive side of the capacitor to the negative differential output, and the negative side of the capacitor to the positive differential output.

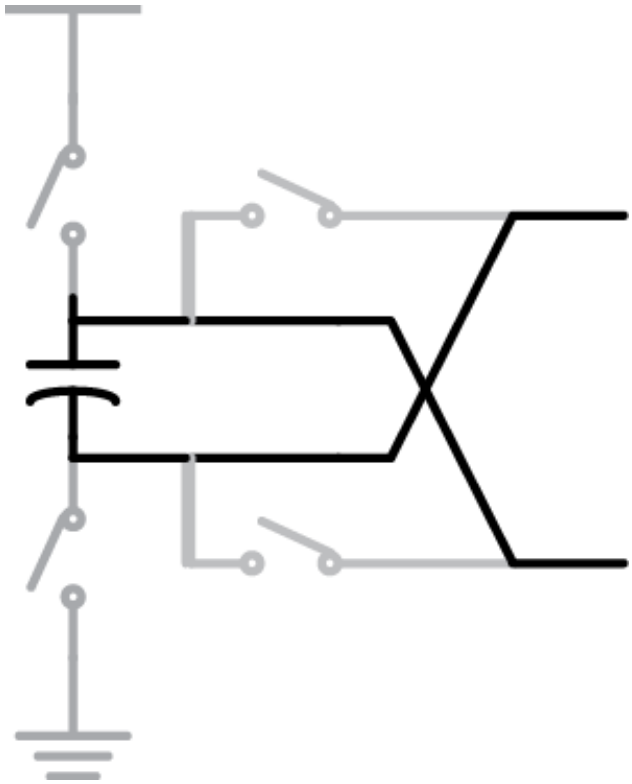


Figure 4.19: TX Minus One

The two phase operation allows a dual data rate (DDR) operation, which has the added benefit of allowing for the introduction of a 3rd state at the output, which is required for the wakeup detector operation. Care needs to be taken in implementing the switched cap cells such that switching in and out of the state of sending data and not does not introduce large spikes on the common mode that would introduce extra latency in the wakeup scheme. A full picture of the tx one could implement is show in fig. 4.20, which introduces swing and impedance control by having multiple parallel segments per phase that can be controlled to change the amount of charge that gets sent to the output network, and conversely the effective resistance seen from the switched cap resistor as well as the series switches.

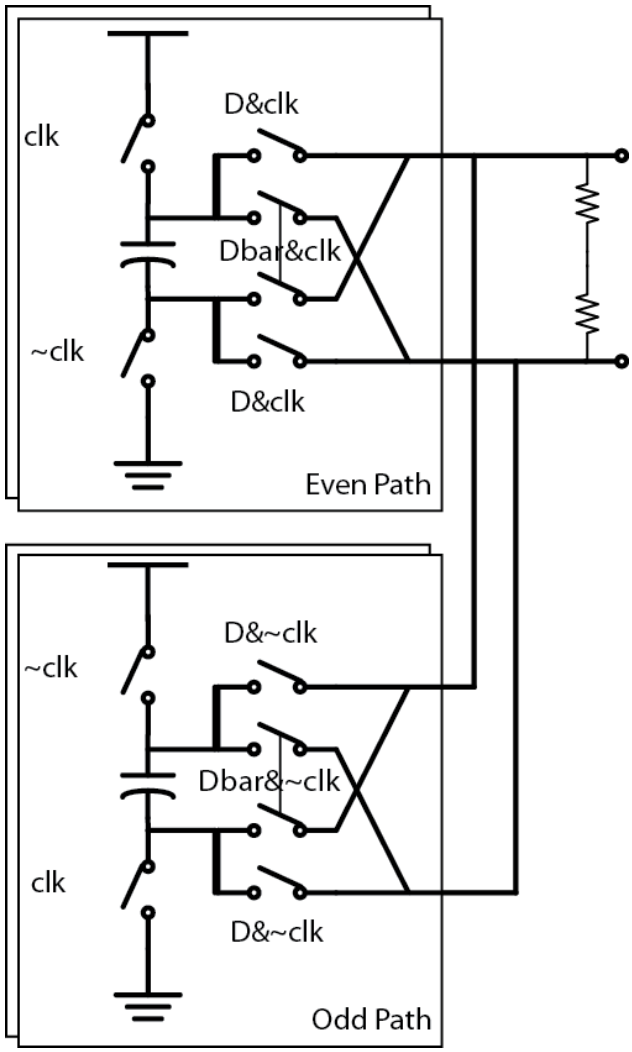


Figure 4.20: TX Detail

An explicit resistor as part of the termination helps ensure some level of matching with the output impedance of the traces across different code settings as well as terminating the common mode, which makes it so that switching between the on and off state does not move the bias point around substantially. This means that the settling time when waking up and subsequently going to sleep is reduced, decreasing the amount of time required to wait for things to settle before being fully operational, which is one of the main system optimizations for burst mode communication systems.

Simulation waveforms from the implementation of this switched capacitor transmitter can be seen in fig. 4.21. This includes the loading of the channel and estimated parasitics of the packaging as well as the post layout extracted versions of the transmitter circuits itself. Due to tapeout time constraints this part of the system was not included in the final design,

so these are the final simulation results for the TX.

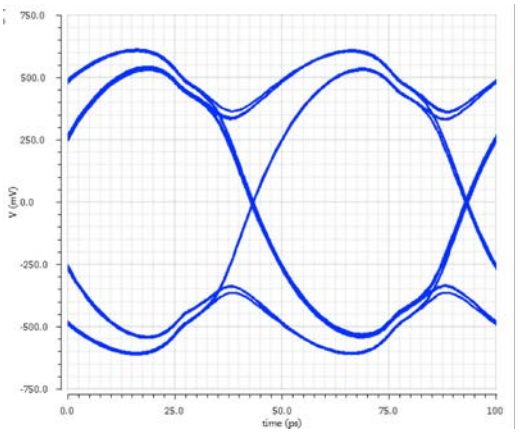


Figure 4.21: Simulated TX Eye Diagram

4.4 Calibration

There are a few main things that need to be calibrated that are standard for links: frequency and phase of the clock as well as equalization coefficients. The schemes for these are similar to the conventional means, so less time will be spent explaining those. Due to the burst mode operation however, there are other settings that need to be calibrated. The clocking is a large majority of those, specifically timing the enable of the system, starting the clock at the right time, verifying the clock is the right frequency, and ensuring that any drift and other variations over time do not stop the system from being operational. On the transmit side of things, the easiest way to do this is to have an enable signal going to the TX cells themselves that is asserted after the clock is up and running, as well as the data has been shifted through the serializer so it is ready for the driver cell as well. This could be done through a controlled delay line and calibrated somehow, but doing so is more complicated than is necessary. Since the blocks mentioned are all clocked, and in the off state the clock is not running you could create a replica of the relevant sections of the serializer and use that as delay elements for this calibration. This same scheme for resetting the sequential elements would ensure that the system would stop after the appropriate amount of time, and currently nothing in the system is being timed on the exact end of the packet in the end to end link case, so the main optimization here is that the circuits burn more power when they are on so you would want to minimize that as much as is easily doable.

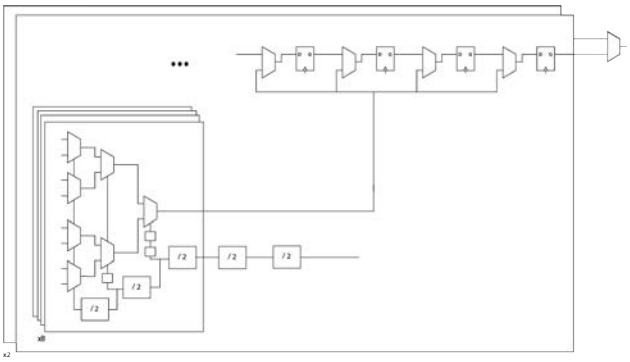


Figure 4.22: Serializer

On the receive side of the link there are a lot more values that need to be set properly because everything is self timed, that is the incoming data stream turns on the receive system. There is some deterministic delay from the incoming data hitting the wakeup detector until the SR latch at its output is asserted, and then some different amount of delay for the oscillator to start up itself, and the system relies on the first 'real' edge of the clock enabling the integrating dynamic latches to match their timing window with the incoming bits. A block diagram of some of the necessary calibration loops to enable this behavior are shown in fig. 4.24, although some of them have been simplified for visual clarity. In order to check if the loops are doing what they are supposed to do, additional resources are required. For example, one of the requirements for turning on the receiver is that the single ended swing coming out of the oscillator has settled, which means that the oscillator is fully on and operational. This requires looking at the analog value of the first edge that the system claims is a real clock edge, which can be done by iterating using the circuit shown in fig. 4.23. This circuit is effectively a sampler triggered on the rising edge of the delayed enable signal, and will compare the clock output to a voltage reference dac that can be swept across multiple packets to find settings that would work in the system. The algorithm that one could employ would be the following - set the delay to the maximum value and sweep the dac voltage codes from low to high until the comparator output changes. Then by stepping back the delay you can figure out what the initial timing was relative to the clock by repeating the procedure. If you have enough codes you can find what the maximum swing is out of the oscillator this way. Then, you can use a code setting that is fraction of the maximum and reduce the delay setting of the delay line until you find the point at which this will no longer operate. That is the minimum delay setting required for the oscillator to start, where in this case starting is defined as the oscillator swing reaching the threshold of the maximum swing that you determined in this calibration procedure. This would calibrate the delay of the oscillator starting up, but has no relation to the incoming data so another loop is required to measure that. However, this is made easier by the fact that you now should have a clock that is working and can borrow some techniques from conventional high speed serial links in the form of a per packet dlev loop to try and maximize the level of the 010 pattern that was

inserted in the header. Before you can guarantee that this will work a frequency calibration should be done to make sure that the oscillation frequency is close enough to the frequency of the incoming data such that accumulating error doesn't introduce extra errors. Since these chips are operating in a system that shares a low frequency reference, a procedure can be done to figure out what the pre programmed multiple of said low frequency clock should be. Once the frequency is calibrated and the delay for the startup of the oscillator itself then the delay from the wakeup detector relative to the data can finally be determined using a sign sign LMS loop based on the dlev of the "1" in the starting 010 on the received data. The last delay setting on the receiver related to this wakeup procedure is that there is a constraint on the relative delay between the clock and enable going to the receiver core compared to the first stage of deserialization, which is the last delay line that again is more straightforward to calibrate and ensure proper operation.

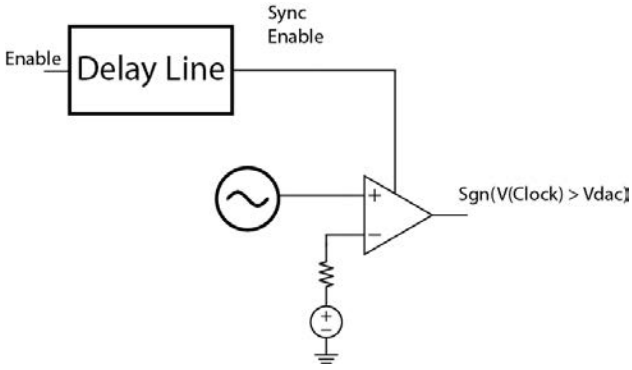


Figure 4.23: Enable Sync Loop

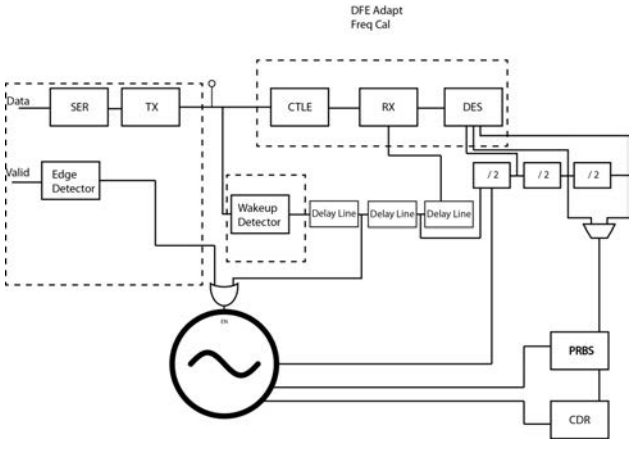


Figure 4.24: Serdes Complete Diagram

4.5 Comparison

As mentioned previously this concept of a burst mode serial link is not entirely new, as there have been other implementations of pieces or the whole thing in the past. A few examples can be found looking at [2, 30, 17] for different pieces of the implementations. A full breakdown of how this work compares to those other designs can be found in table 4.1.

	This Work	[2]	[30]	[17]
Technology	16nm	65nm GP	40nm LP	40nm LP
Data Rate (Gb/s)	20	7	2.5-5.6	2.7-4.3
Startup Time (ns)	< 2	< 20	8	241.8
Energy Efficiency (pJ/b)	2.5	9.1	2.4	3.3
On-state Power (mW)	50	63.7	13.4	14.2
Off-state Power (mW)	800	740	0 *	50
De/ Serialization Ratio	256:1	16:1	N/A	8:1
Output Swing (mV Diff pk-pk)	700	500	N/A	200

Table 4.1: Comparison to prior art for building burst mode serial links

4.6 Learnings

The prior sections describe the steps that would be required in order to design and build an energy efficient and burst mode serial link system end to end. This includes the necessary configuration and calibration to ensure correct timing operation of the entire system during operation, as well as equalization required by the channel for proper operation at a desirable bit error rate. The die photo of the implemented system can be seen in fig. 4.25, however due to limitations in time and testing setup chip results were not obtained. For time restrictions the transmit half of the system was not included, so this chip includes the receive datapath, clocking, wakeup detector, and calibration loops. Due to limitations in the testing and debugging interface data was not obtained from the chip, and the cause was not determined.

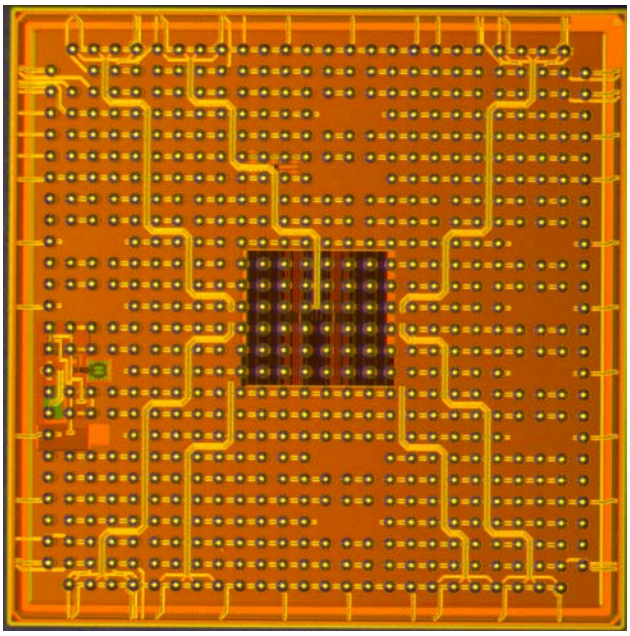


Figure 4.25: System Die Photo

Chapter 5

Conclusion

To conclude, limit the amount of data and make energy efficient wired and wireless links that only spend power to send bits, and spend absolutely zero power otherwise. In the context of a system designed for reading neural activity out of an implanatable system, a compression algorithm is proposed to greatly reduce the amount of data required to be sent. The structure is such that for the applications that are being targeted there is little to no reduction in system level fidelity, with large decreases in the amount of data being sent. Because this is in a system where the data transfer is a large power consumer, overall the system itself can be much more efficient. Once the data has been reduced as much as is easily possible within the specific application the data still needs to be transmitted to where it needs to be used. General purpose solutions for making that transfer efficient can only have so much progress, and instead a more application specific approach can also be used to target that part of the system.

Two different but similar schemes are introduced as to how to design and build circuits that would achieve this, as well as a way to compress the data in an application specific way that drastically reduces the data rate in a way that does not negatively impact the overall system performance. In these schemes a peak data rate is supported with minimal latency, and there can be a tradeoff between latency and data rate when backing off from that peak data rate. In the wired link context a low latency can be guaranteed across a wide range of data rates simply by ensuring that the packets are sent as quickly as they arrive. This enables the system to use the information that it knows when there is data to be sent, and can enter a much lower power state when that data is not being transferred. Since both sides of the link can share a common low frequency reference, a wakeup type scheme can allow packets to show up at calibrated times that can be determined with a minor amount of overhead. The wireless link has the limitation that a more constant data rate needs to be sent to maintain the tracking of the link itself, but further work could be done to adopt a similar scheme if there was an application that warranted this approach. It however still can use a similar scheme to continuously update its tracking of not only the data being sent but system level parameters such as a beamforming matrix all in mostly real time with minor overhead due to calibration.

With these techniques and details about the calibration loops necessary to design these systems one can design more efficient systems targeted to the specific applications by using the extra information about those systems instead of simply using general purpose components. The ideas do not have to be restricted to the targeted applications shown, but instead should be seen as strategies that can be used to target other applications where system level information can change the rate and nature of data transfer. With the trend of a large amount of analog and mixed signal circuits moving more heavily into the digital or at least digitally assisted domain, some of the techniques presented can even be applied more broadly to other types of circuits in various other applications.

Bibliography

[1] D. Abts et al. "Energy proportional datacenter". In: *ACM ISCA*. 2010.

[2] T. Anand et al. "3.7 A 7Gb/s rapid on/off embedded-clock serial-link transceiver with 20ns power-on time, 740μW off-state power for energy-proportional links in 65nm CMOS". In: *2015 IEEE International Solid-State Circuits Conference - (ISSCC) Digest of Technical Papers*. 2015, pp. 1–3. DOI: 10.1109/ISSCC.2015.7062927.

[3] T. Anand et al. "A 7 Gb/s Embedded Clock Transceiver for Energy Proportional Links". In: *IEEE Journal of Solid-State Circuits* 50.12 (2015), pp. 3101–3119.

[4] Meysam Azin et al. "A battery-powered activity-dependent intracortical microstimulation IC for brain-machine-brain interface". In: *Solid-State Circuits, IEEE Journal of* 46.4 (2011), pp. 731–745.

[5] W. Biederman et al. "A 4.78 mm 2 Fully-Integrated Neuromodulation SoC Combining 64 Acquisition Channels With Digital Compression and Simultaneous Dual Stimulation". In: *IEEE Journal of Solid-State Circuits* 50.4 (2015), pp. 1038–1047. DOI: 10.1109/JSSC.2014.2384736.

[6] Moosung Chae et al. "A 128-Channel 6mW Wireless Neural Recording IC with On-the-Fly Spike Sorting and UWB Tansmitter". In: *Solid-State Circuits Conference, 2008. ISSCC 2008. Digest of Technical Papers. IEEE International*. 2008, pp. 146 –603. DOI: 10.1109/ISSCC.2008.4523099.

[7] Tung-Chien Chen et al. "A biomedical multiprocessor SoC for closed-loop neuroprosthetic applications". In: *Solid-State Circuits Conference-Digest of Technical Papers, 2009. ISSCC 2009. IEEE International*. IEEE. 2009, pp. 434–435.

[8] K. Fukuda et al. "A 12.3mW 12.5Gb/s complete transceiver in 65nm CMOS". In: *2010 IEEE International Solid-State Circuits Conference - (ISSCC)*. 2010, pp. 368–369. DOI: 10.1109/ISSCC.2010.5433824.

[9] R. R. Harrison et al. "A Low-Power Integrated Circuit for a Wireless 100-Electrode Neural Recording System". In: *Solid-State Circuits, IEEE Journal of* 42.1 (2007), pp. 123 –133. ISSN: 0018-9200. DOI: 10.1109/JSSC.2006.886567.

[10] Reid R Harrison et al. "Wireless neural recording with single low-power integrated circuit". In: *Neural Systems and Rehabilitation Engineering, IEEE Transactions on* 17.4 (2009), pp. 322–329.

[11] R.R. Harrison and C. Charles. "A low-power low-noise CMOS amplifier for neural recording applications". In: *Solid-State Circuits, IEEE Journal of* 38.6 (2003), pp. 958–965.

[12] Thomas Jochum, Timothy Denison, and Patrick Wolf. "Integrated circuit amplifiers for multi-electrode intracortical recording". In: *Journal of Neural Engineering* 6.1 (Feb. 2009), p. 012001.

[13] Vaibhav Karkare, Sarah Gibson, and Dejan Markovic. "A 130-μW, 64-channel spike-sorting DSP chip". In: *Solid-State Circuits Conference, 2009. A-SSCC 2009. IEEE Asian*. IEEE. 2009, pp. 289–292.

[14] L. Kong, D. Seo, and E. Alon. "A 50mW-TX 65mW-RX 60GHz 4-element phased-array transceiver with integrated antennas in 65nm CMOS". In: *2013 IEEE International Solid-State Circuits Conference Digest of Technical Papers*. 2013, pp. 234–235. DOI: 10.1109/ISSCC.2013.6487714.

[15] Hyunjung Lee et al. "Biomechanical analysis of silicon microelectrode-induced strain in the brain". In: *Journal of Neural Engineering* 2.4 (2005), p. 81.

[16] Seung Bae Lee et al. "An Inductively Powered Scalable 32-Channel Wireless Neural Recording System-on-a-Chip for Neuroscience Applications". In: *Biomedical Circuits and Systems, IEEE Transactions on* 4.6 (2010), pp. 360 –371. ISSN: 1932-4545. DOI: 10.1109/TBCAS.2010.2078814.

[17] B. Leibowitz et al. "A 4.3 GB/s Mobile Memory Interface With Power-Efficient Bandwidth Scaling". In: *IEEE Journal of Solid-State Circuits* 45.4 (2010), pp. 889–898. DOI: 10.1109/JSSC.2010.2040230.

[18] R. Muller, S. Gambini, and J.M. Rabaey. "A 0.013mm^2, 5μW, DC-Coupled Neural Signal Acquisition IC With 0.5V Supply". In: *Solid-State Circuits, IEEE Journal of* 47.1 (2012), pp. 232 –243. ISSN: 0018-9200. DOI: 10.1109/JSSC.2011.2163552.

[19] R. Palmer et al. "Design considerations for low-power high-performance mobile logic and memory interfaces". In: *2008 IEEE Asian Solid-State Circuits Conference*. 2008, pp. 205–208. DOI: 10.1109/ASSCC.2008.4708764.

[20] J. W. Poulton et al. "A 0.54 pJ/b 20 Gb/s Ground-Referenced Single-Ended Short-Reach Serial Link in 28 nm CMOS for Advanced Packaging Applications". In: *IEEE Journal of Solid-State Circuits* 48.12 (2013), pp. 3206–3218. ISSN: 0018-9200. DOI: 10.1109/JSSC.2013.2279053.

[21] A. Puglielli et al. "A scalable massive MIMO array architecture based on common modules". In: *2015 IEEE International Conference on Communication Workshop (ICCW)*. 2015, pp. 1310–1315.

[22] A. Puglielli et al. "Design of Energy- and Cost-Efficient Massive MIMO Arrays". In: *Proceedings of the IEEE* 104.3 (2016), pp. 586–606. DOI: 10.1109/JPROC.2015.2492539.

[23] Hyo-Gyuem Rhew et al. "A wirelessly powered log-based closed-loop deep brain stimulation SoC with two-way wireless telemetry for treatment of neurological disorders". In: *VLSI Circuits (VLSIC), 2012 Symposium on*. IEEE. 2012, pp. 70–71.

[24] A.M. Sodagar et al. "An implantable 64-channel wireless microsystem for single-unit neural recording". In: *Solid-State Circuits, IEEE Journal of* 44.9 (2009), pp. 2591–2604.

[25] N. Sutardja et al. "A 2-tap switched capacitor FFE transmitter achieving 1-20 Gb/s at 0.72-0.62 pJ/bit". In: *ESSCIRC 2019 - IEEE 45th European Solid State Circuits Conference (ESSCIRC)*. 2019, pp. 273–276.

[26] W. Wattanapanitch, M. Fee, and R. Sarpeshkar. "An energy-efficient micropower neural recording amplifier". In: *Biomedical Circuits and Systems, IEEE Transactions on* 1.2 (2007), pp. 136–147.

[27] W. Wattanapanitch and R. Sarpeshkar. "A Low-Power 32-Channel Digitally Programmable Neural Recording Integrated Circuit". In: *IEEE Transactions on Biomedical Circuits and Systems* 5.6 (Dec. 2011), pp. 592 –602. ISSN: 1932-4545. DOI: 10.1109/TBCAS.2011.2163404.

[28] D. Wei et al. "A 10-Gb/s/ch, 0.6-pJ/bit/mm Power Scalable Rapid-ON/OFF Transceiver for On-Chip Energy Proportional Interconnects". In: *IEEE Journal of Solid-State Circuits* 53.3 (2018), pp. 873–883.

[29] Johan Wessberg et al. "Real-time prediction of hand trajectory by ensembles of cortical neurons in primates". In: *Nature* 408.6810 (2000), pp. 361–365.

[30] J. Zerbe et al. "A 5.6Gb/s 2.4mW/Gb/s bidirectional link with 8ns power-on". In: *2011 Symposium on VLSI Circuits - Digest of Technical Papers*. 2011, pp. 82–83.