Gabriel Fellner

High-Speed Peak Detect and Hold Circuit for ESD Threat-Level Analysis of Portable Devices

Master's Thesis

to achieve the university degree of

Master of Science

Master's degree programme: Electrical Engineering

submitted to

Graz University of Technology

Supervisor

Univ.-Prof. Dipl.-Ing. Dr.-Ing. David Johannes Pommerenke

Institute of Electronics
Head: Univ.-Prof. Dipl-Ing. Dr.techn. Bernd Deutschmann

Graz, May 2021

Kurzfassung

Da elektrostatische Entladungen (ESD) eine bekannte Bedrohung für elektronische Geräte darstellen, ist es sehr wichtig Informationen über die Spannungs- und Strompegel und deren Auftretensrate zu sammeln. Diese Informationen ermöglichen es den Elektronikentwicklern die Geräte mit einem geeigneten ESD-Schutz zu versehen, um die Zuverlässigkeit im Feld zu gewährleisten.
Zum Zeitpunkt dieser Arbeit gab es bereits Forschungsstudien, die Statistiken mit diesen Informationen für geerdete Geräte liefern, aber es gibt keine Informationen über die ESD-Bedrohungspegel für batteriebetriebene (nicht geerdete) Geräte. Aufgrund der Tatsache, dass batteriebetriebene Geräte immer mehr an Bedeutung gewinnen, wie zum Beispiel Mobiltelefone, Tablet-Computer, Smartwatches und viele mehr, besteht ein Bedarf an einer Bedrohungsgradanalyse, die die ESD-Belastung aufzeigt, die in der Realität auf diese Geräte einwirkt.
Für diese Analyse wurde ein neues Systemkonzept entwickelt, welches die elektromagnetischen Felder während eines ESD-Ereignisses misst und diese Daten mit Referenzdaten in Beziehung setzt, die in einem IEC 61000-4-2 konformen Testaufbau generiert wurden. Aufgrund der hohen Geschwindigkeit der Feldstärken während eines ESD-Ereignisses wird nur der Spitzenwert der Signale ausgewertet.
Gegenstand dieser Arbeit ist der Entwurf einer Peak-Detect-and-Hold Schaltung, die es ermöglicht, den Spitzenwert von sehr schnellen und schmalen transienten Spannungsimpulsen, die von den verwendeten Sensoren erzeugt werden, zu erfassen. Aufgrund des batteriebetriebs der tragbaren Geräte stellt der Stromverbrauch eine wesentliche Einschränkung für das Analysesystem dar. Einer der Hauptvorteile der vorgestellten Schaltung ist der per Software einstellbare Dynamikbereich, der den Stromverbrauch durch Anpassung der PDH-Spezifikation an die Spezifikation des Sensorausgangs optimieren kann. Mit der vorgestellten Schaltung wurde ein Dynamikbereich von 37dB für Pulsbreiten von 3ns bei nur 108mW Leistungsaufnahme und ein Dynamikbereich von 26,7dB für Pulsbreiten von 1,5ns bei einer Leistungsaufnahme von 153mW erreicht.

Abstract

Since electrostatic discharge (ESD) is a known threat for electronic devices it is very important to gather information about the voltage- and current-levels and their rate of occurrence. This information enables electronic design engineers to add appropriate ESD-protection to devices to ensure reliability in the field. At the time of this thesis there were already research studies that provide statistics with this information for grounded devices, but there is no information about the ESD threat levels for battery-powered (non-grounded) devices. Due to the fact that battery-powered devices are becoming increasingly important, for example cell phones, tablet-computers, smart watches and many more, there is a need for a threat level analysis which reveals the ESD-stress which is applied to those devices in reality.

For this analysis a new system concept was created which measures the electro-magnetic fields during an ESD event and relates this data to reference data, generated in an IEC 61000-4-2 standard compliant test setup. Due to the inherent speed of the field strengths during an ESD event only the peak value of the signals is analysed.

The scope of this work is the design of a peak detect and hold circuit which allows capturing the peak value of very fast and narrow transient voltage pulses generated by the sensors used. Because of the battery powered operation of the portable devices the power consumption is a major limitation for the analysis system. One of the key benefits of the presented circuit is the software adjustable dynamic range which is able to optimize power-consumption by matching the PDH performance to the sensor output specification. With the presented circuit a dynamic range of 37dB for pulse-widths of 3ns was achieved with only 108mW power consumption and a dynamic range of 26.7dB for pulse-widths of 1.5ns by a power consumption of 153mW.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Electronics industry is putting large effort in shrinking the structure size of transistors year by year and during the time of this thesis already ICs with structure size as small as 7nm are in serial production and available in consumer devices. The shrinking of the minimum transistor size made it possible to get incredible computational performance available to very favorable prices, driving innovations and research of the whole world. However, with the gain in performance of these very small structures came an increased susceptibility to electrostatic discharge (ESD) events. Testing the susceptibility of devices to ESD is mandatory and the IEC 61000-4-2 therefor defines the test setup to use. Although the test levels are defined in this standard this does not mean that the ESD voltages for which is tested are voltage levels appearing in reality. For grounded devices there are scientific studies reveling the ESD-levels appearing in reality providing information on the rate of occurrence for different voltage levels and a probability distribution function of the voltages [1] [2]. For non-grounded (floating) devices there is no such study at the time of this thesis. Since portable devices are increasing in importance due to cell phones, tablet-computers, laptops and smart watches there is a need for such a study on non-grounded devices.
This thesis is part of a project dealing with the question of ESD threat-levels for portable devices. The goal is to provide a scientific study on the appearance of ESD on portable devices as cell phones. The rate of occurrence for certain voltage levels and a probability distribution function should be created, allowing engineers the design of hardware protected against ESD up to a certain probability.
For collecting the necessary data a whole new system concept was designed which should relate sensor data to ESD test levels in the test setup defined by the standard. One of the major difficulties in this measurement is the fact that the devices are usually not grounded, therefore different electro-magnetic field sensors are used which capture the appearing field strengths during an ESD-event. This sensor data should be used for estimating the ESD discharge voltage by the use of measured and simulated reference data.
Because of the inherent speed of the fields generated during such an discharge event and the limiting factor of battery operation, not the whole time domain response of the sensors is used, but only the peak values. This should still enable an estimation of the maximum threat of a discharge event.

There is a Major difference between ESD to grounded and non-grounded devices. In grounded devices there is an approximately known return-path for the current, which directly relates the ESD-stress of the device to the ESD-voltage. Whereas in non-grounded devices there are many different configurations for the current return-path, making a direct relation between ESD voltage and ESD-stress nearly impossible. Therefore different scenarios which are relevant for portable devices are analyzed in the following section.

1.1 ESD Scenarios for Portable Devices

Thinking of portable battery powered devices there are many different scenarios where electrostatic discharge can occur. The different scenarios also have a different effect on the device, and also with the same ESD voltage some scenarios are a much more severe threat to the device than others. The key parameters which are considered are the impedance seen by the discharge and the capacitance of the ESD source. The most relevant cases, which have very different effect on the severity of the event to the device are explained here.

1.1.1 Scenario 1

Cell phone placed on grounded metallic surface and charged person touches cell phone. Just imagine you are sitting due to COVID-19 pandemic at home for writing your master thesis, just beside your desk there is a wine refrigerator with metallic enclosure, where you most of the time place your phone on. Due to the fact that you have an office chair which is made of polyester and this chair is placed on a insulating PVC mat which should protect your parquet floor, the setup for an ESD generator is ready. Now every time you wiggle around on your chair you charge up and if you reach for your phone an electro-static discharge takes place.

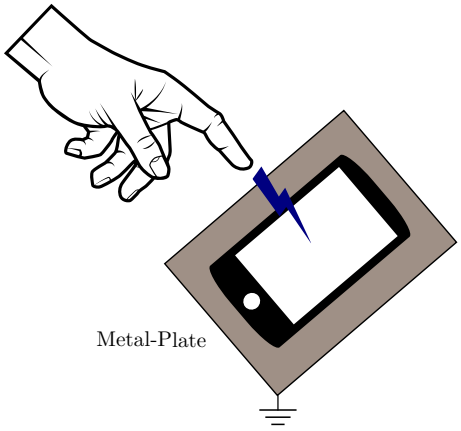


Figure 1.1: ESD Scenario 1

Severity of scenario 1

In this scenario the path to ground for the ESD event has very low impedance (high capacitance or maybe even ohmic contact). This results in a very high discharge current which is a very big threat to the device because of the large electro-magnetic field strengths.

1.1.2 Scenario 2

Charged person holds cell phone and touches grounded conductive surface with cell phone. If you for example hold your cell phone in one hand while walking across a synthetic carpet with shoes that have a rubber sole and you accidentally touch a radiator with your phone. Because of the friction due to walking on the synthetic carpet with your shoes charge carriers are separated and your body is charged to a certain potential. In the moment the grounded radiator is touched the ESD happens.

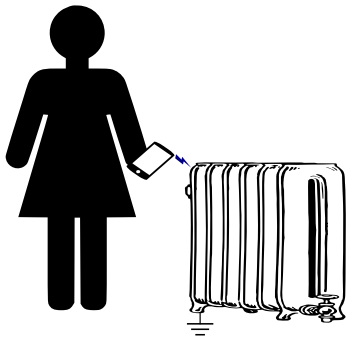


Figure 1.2: ESD Scenario 2

Severity of Scenario 2

This scenario is very similar to scenario 1, but only inverse. So also this scenario is a big threat for the device because the path to ground is again very low impedance, again resulting in very high discharge currents which endanger the device.

1.1.3 Scenario 3

Charged person touches cell phone inside a car, while cell phone is connected to power in the car.
A scenario which is very likely to happen if you have poor geography knowledge but still want to go on holiday by car and navigate by cell phone (somehow me).

Severity of Scenario 3

Inside a car there is the special case, that the capacitance of the human body to the chassis is much higher than the capacitance of a human body inside a room. The reason for the increased capacitance is that the car chassis is conductive and all around the body in a quite small distance. This high capacitance results in a discharge with high energy, so a longer discharge pulse.

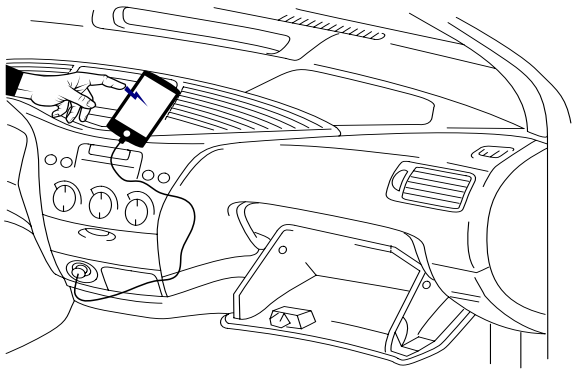


Figure 1.3: ESD Scenario 3

1.1.4 Scenario 4

Charged person touches cell phone placed on insulating surface.
One example for this scenario would be that a person is charged up and then touching the cell phone which is placed on a wooden table.

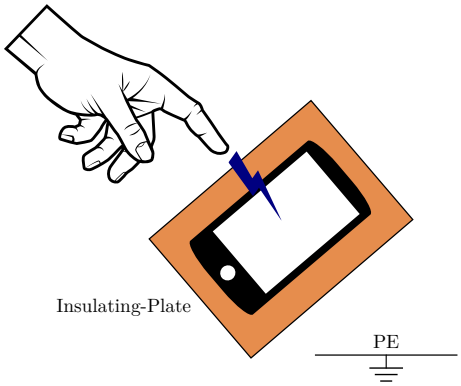


Figure 1.4: ESD Scenario 4

Severity of Scenario 4

Here there is the advantage, that there is no ohmic connection from the device to ground, but only a capacitance from the device to ground. This fact would result in a discharge current that is strongly dependent on the capacitance from the device to ground. If this capacitance is very high again a high peak discharge current will occur, but only for a small amount of time. This is because the device is now able to change its potential and therefore the current decreases when the potential of the device increases.

1.1.5 Scenario 5

Cell phone in one hand and touching with the other hand.
This ESD scenario is very likely to happen during the use of a portable device. When wearing synthetic cloths it can happen, that there is a potential difference between both

hands. If the phone is now held in one hand and touched with the other hand a discharge is happening which balances the charges.

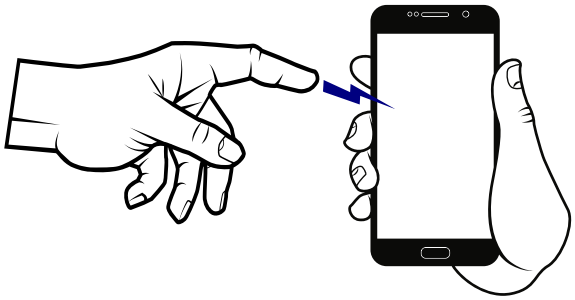


Figure 1.5: ESD Scenario 5

Severity of Scenario 5

Because the human body is slightly conductive the voltage in this case is limited to very low voltages. This low voltage together with the high resistance of the human body results in a very low ESD-stress for the device.

1.2 Equivalent Discharge Voltage

Because the severity of the ESD events dose not only depend on the discharge voltage but also on the discharge scenario, we introduce an equivalent discharge voltage. This equivalent discharge voltage is the voltage applied to the device using the setup described in the IEC 61000-4-2 (setup for non-grounded devices) which results in the same ESD-stress as the real voltage in different discharge scenarios. This eliminates the effect of the different discharge scenarios and enables a meaningful statistic on discharge voltage and rate of occurrence. Introducing this equivalent discharge voltage makes this statistic even more useful for practical applications, because the ESD voltages measured in reality are directly transferred to the standardized setup of the IEC which is used anyway.

1.3 ESD Threat Level Analysis System

For collecting the required data for a threat level analysis, a system has to be developed, which is capable of measuring the voltage and the corresponding current of an ESD event. The measurement results should be stored together with other environmental information and from time to time transferred either to the device itself or to another read-out device. This system should be mounted to the device under test (DUT) or directly integrated in the case of the device. Because the goal of the project is to measure the ESD-stress to battery powered non-grounded devices, also the system has to be battery-powered, either by an own battery or powered by the DUT.

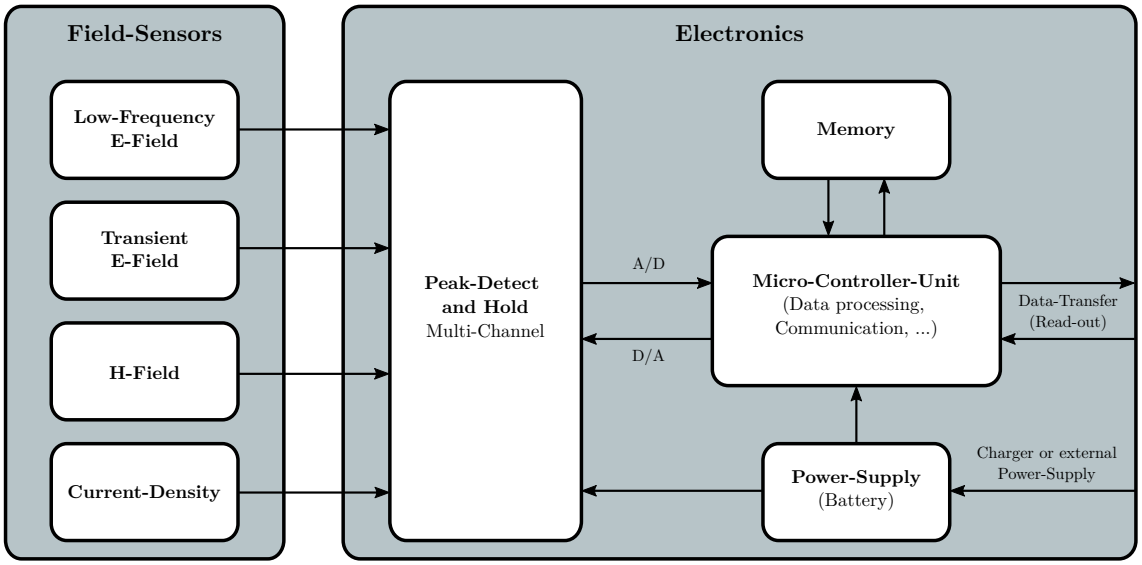


Figure 1.6: ESD Threat Level Analysis System Block Diagram

1.3.1 Measurement Principle

The most difficult thing in measuring the discharge voltage is that the device is not connected to ground, so the absolute value of the discharge voltage can not be measured directly. Also information about the abslolut value of the voltage would not provide the whole information about the threat level, because of the different discharge scenarios. For this reason the equivalent discharge voltage was introduced above.

The approach that is used in this project is to measure the electric- and magnetic-field during the ESD-event and correlate this with measurement data from reference measurements and reference full-wave simulations. Using the setup from the IEC standard, reference data can be generated by testing the portable device with different voltage levels. During this reference measurements the field strength for different discharge voltages is saved as reference data. Same can be done for full-wave simulation.
By using this reference data, the essential information that is generated, is that one can say for example, an ESD event during use caused very similar field strengths as a XkV discharge in the standardized setup. Where the electromagnetic-fields are defined as ESD stress.
Because the measurement of this fast transient signals is very challenging and probably the most information is stored in the peak value of the signals a peak detect and hold circuit is used. This also reduces required computation power which would be necessary when the whole pulse would have to be processed and analyzed.

1.3.2 Sensors

For the measurement of the fields different sensors are used:

- Low-frequency E-field sensor

- Transient E-field sensor
- H-field sensor
- Current density sensor

The low-frequency E-field sensor is used to measure slow changes in the static electric field which are likely to occur before an ESD event. Whereas the transient E-field sensor measures fast changes in the electric field which are usually produced during the discharge event. Both sensors should provide information about the discharge voltage.
For gaining information about the discharge current the H-field sensor and the current density sensor are used. Four different types of sensors are used, because for each sensor output only the peak voltage is measured and using four different types of sensors should give more information about the shape of the signal. Getting information from these four different sensors should enable a correlation between measured reference data and real measurement data.

1.3.3 Peak Detection

Electro static discharges are very fast events, their rise-time is only limited by inductance formed by the loop of the discharge current. In the IEC61000-4-2 the current rise-time is specified with 0.8ns but in reality this value can vary depending on the discharge scenario. This fast rise-time together with only a short duration of this initial peak of about 4ns brings very tough requirements for a peak detect and hold circuit.
The goal is to use a peak detect and hold circuit which holds the peak value of the sensor output, so that it is possible to read this value with a slow ADC to process and store the data.

1.3.4 Data Processing

For the analog to digital conversion an integrated ADC of a micro controller unit is used. This digital data is processed and stored to the internal flash together with a time-stamp of the event and other sensor data. The micro controller is also able to adjust the DC-operating point of the peak detect and hold circuit by using external DACs. This adjustment of the bias point enables a circuit adjustment of each peak detect and hold circuit to the required specifications of the connected field sensor.

1.4 Thesis Objectives

This thesis is part of the project explained above which focuses on the ESD threat level analysis for portable devices. The scope of this thesis is on the part of designing a peak detect and hold circuit, which is suitable to be used in this application.

Meaning that an appropriate topology has to be found that is able to detect this fast transient voltage peaks with rise-times in the sub nano-seconds range and very narrow pulse-widths. Another concern is dynamic range, because ESD voltages can vary from 500V and even smaller up to 25kV resulting also in a huge measurement range of this peak detect and hold circuit. Because the portable devices where the system should be used

are quite small and suffer itself from limited battery, this circuit should have low power consumption. The low power consumption then enables the system to be mounted on the portable device with either a small battery for the system itself or to be powered by the portable device. Of course the small form factor is also a criteria for the peak detector.

Summarized the requirements for this peak detect and hold circuit are:

- Detect pulses with rise-times in the range of 0.8ns and pulse-widths of 4ns
- High dynamic range, 34dB for ESD voltage range from 500V to 25kV
- Low power consumption, battery-operation must be possible
- Small form factor, integration or mounting on portable device like cellphone should be possible without affecting usability of device

In the course of this master thesis different circuit concepts were designed, simulated and built. At first important parameters of the input signal were defined which may have effect on the result of the peak detection. Focusing on this key parameters of the input signal a simulation test bench was created which enables testing and optimizing before starting the manufacturing process. This simulation setup allows a first estimation of the circuit response to different input signals, considering different rise-times, pulse-widths and amplitudes.
For real performance tests a test setup was generated that is able to perform the same analysis as in simulation with the real circuit.

Chapter 2

Literature Research

In this chapter an overview is given on existing methods for detecting peak values of voltage signals. The focus of this chapter is on discrete circuits using operational amplifiers or bipolar transistors as well as integrated circuits using complementary metal-oxide-semiconductor (CMOS) technology. For reasons of completeness also the basic structures are presented.

2.1 Peak Detect and Hold Principle

The operating principle of a peak detection and hold circuit (PDH) is, that the circuit generates an output voltage which is equal to the maximum of the input signal. This functionality is in general implemented by a rectifying element which only allows current flow in one direction followed by a memory element, which is able to store the voltage. The rectifying element can be for example a diode (but also a voltage controlled switch) and the memory element is usually a capacitor. It is also possible to bring the signal into the digital world by an analog to digital converter (ADC) and perform the peak detection in the digital domain.

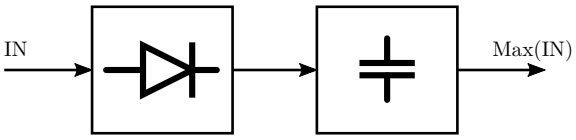


Figure 2.1: Peak Detect and Hold Principle

2.2 Overview of Basic Peak Detect and Hold Circuits

2.2.1 Diode capacitor configuration PDH

The simplest form of a PDH is a series configuration of a diode and a capacitor as shown in figure 2.2 . If a rising signal is applied to this circuit, the diode is forward biased and starts charging up the capacitor, when the input voltage starts decreasing the diode becomes reverse biased and the voltage at the capacitor stays constant. This configuration has

significant drawbacks since the input impedance is variable and it is not usable for input voltages smaller than the built in voltage of the diode. Furthermore the voltage drop of the diode depends on the temperature and the current through the diode, so the accuracy is strongly affected by the rise time of the input signal [3].

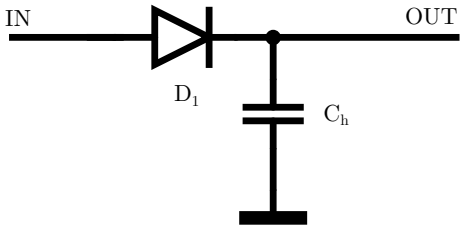


Figure 2.2: Diode Capacitor PDH

2.2.2 Operational Amplifier PDH

By the use of operational amplifiers a better circuit can be created, which provides a high input impedance and also works for small input voltages. The reduction in the output voltage which results from the diode drop can be eliminated by providing the feedback for the amplifier directly from the capacitor. If the input signal is higher than the voltage at the hold capacitor, the capacitor is charged to the input voltage. When the input voltage is smaller than the voltage at the capacitor the output of the operational amplifier drops to the negative supply rail.

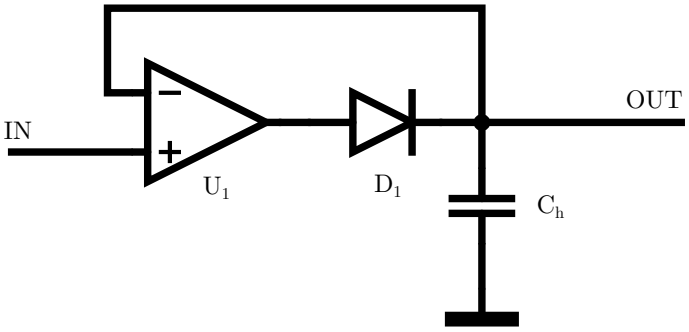


Figure 2.3: Operational Amplifier PDH

In this configuration the parameters of the operational amplifier mainly determine the behavior. The most important limitations come from the finite slew-rate, the input bias current and the maximum output current of the amplifier. Hence the output of the amplifier drops to the negative supply rail during hold state, the output of the amplifier has to increase from the negative supply rail to the input voltage plus the diode drop when the capacitor has to be charged. This is especially critical when there is a small pulse which brings the circuit to hold mode just before the real pulse which should be detected. This big drop of the output voltage when changing to hold mode also discharges the hold capacitor because of the parallel capacitance of the diode[3][4]. Especially for detecting very narrow pulse-widths the hold capacitor should be very small (in the pF-range) resulting in a large

voltage error due to cross talk through the diode capacitance. Also the dynamic range is limited by this voltage error.

2.2.3 Improved Operational Amplifier PDH

The circuit shown in figure 2.4 is very similar to the operational amplifier PDH shown above. In the peak detection mode when the input voltage is rising the hold capacitor is charged through the diode D_1 until the output voltage of the second operational amplifier in voltage follower configuration is equal to the input voltage. In this peak detection phase the diode D_2 is reverse biased. When the input voltage starts to decrease, the diode D_1 gets reverse biased and the output voltage of the first operational amplifier starts to decrease, but is clamped to one diode drop below its input voltage because D_2 starts conducting. This brings the big advantage that the output of the first opamp does not drop to the negative supply rail. Resulting in lower reverse bias voltage for the diode D_1 and therefore less leakage current and also less discharge of the hold capacitor due to the diode capacitance. Also a change from the hold phase to the peak detection phase requires here a lower slew-rate because of the smaller voltage increase required [4].

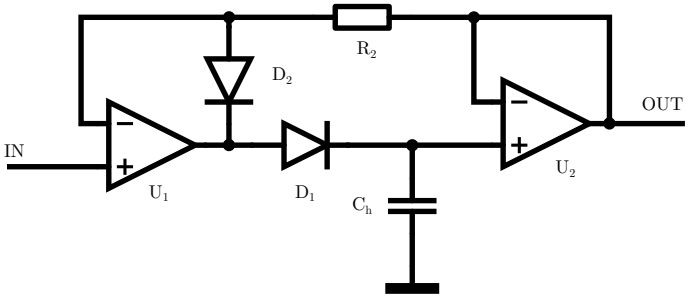


Figure 2.4: Improved Operational Amplifier PDH

2.2.4 Sampling Operational Transconductance Amplifier PDH

For detecting the peak voltage it is also possible to use a sampling transconductance amplifier (SOTA). The basic operation of the OTA is that the input voltage difference is converted to an output current which is used for charging the hold capacitor. Also the above explained circuits with operational amplifiers can be used with OTA instead of operational amplifier. Using a SOTA brings the advantage that no diode is necessary because when the peak voltage is reached at the hold capacitor the output of the SOTA can be turned off. For the turn-off of the SOTA output a control signal must be generate that triggers the SOTA when the peak voltage is reached. Generally there are two ways for generating this trigger signal, either leading edge discrimination (LED) or a constant fraction discriminator (CFD).

2.2.5 Continuous Sampling ADC

Using an ADC which is continuously sampling would seem as the easiest solution. But for detecting very fast transient signals a very high sampling rate is required. Especially if the voltage waveform shape is a very narrow peak depending on the sampling frequency it

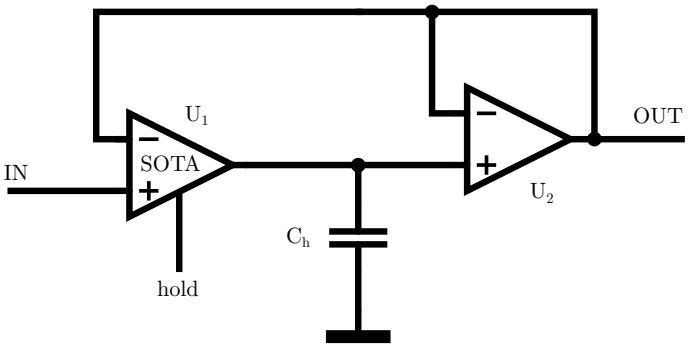


Figure 2.5: SOTA PDH

can happen that the peak voltage is between two sampling points. If we assume a required sampling period of 2ns this would result in 500MS/s, which is also a challenge for the system that has to process this data for finding voltage peaks.

2.2.6 Triggered Sampling ADC

Using an ADC which is triggered by a control signal generated by LED or CFD is also possible to detect and hold peak voltages. In this case there is some uncertainty in the timing of the trigger signal, specially when the exact shape of the signal is not known, it is very difficult to generate a trigger signal at the voltage peak. Also the propagation delay which is needed by the ADC to generate a sample is a limiting factor, when the input signal is fast.

2.2.7 Digital PDH

Thinking on the structure of a flash ADC it is also possible to make a very similar structure as peak detection circuit. This digital peak detect and hold circuit directly converts the input peak voltage to a digital output signal, so in this case there is no need for analog to digital conversion. In this structure comparators compare the input voltage to a given voltage threshold, if this threshold is exceeded by the input signal the output of the comparator changes to „high" and triggers the set input of an RS-Latch. The peak value of the input signal is given in thermometer-code and can be converted to binary-code using a priority encoder. A discrete digital PDH with 12ns conversion time was implemented and described in [5].
Propagation delay time of the comparator and the latch are the key parameter of this topology, because they define the minimum detectable pulse width. Instead of a RS-Latch also a D-Flipflop with reset can be used, these components are more available with faster timing specifications. To achieve a reasonable accuracy the component count is very high and therefore also the required area of the circuit and power consumption. Because of the high component count this topology would be very suitable for integrated circuits.

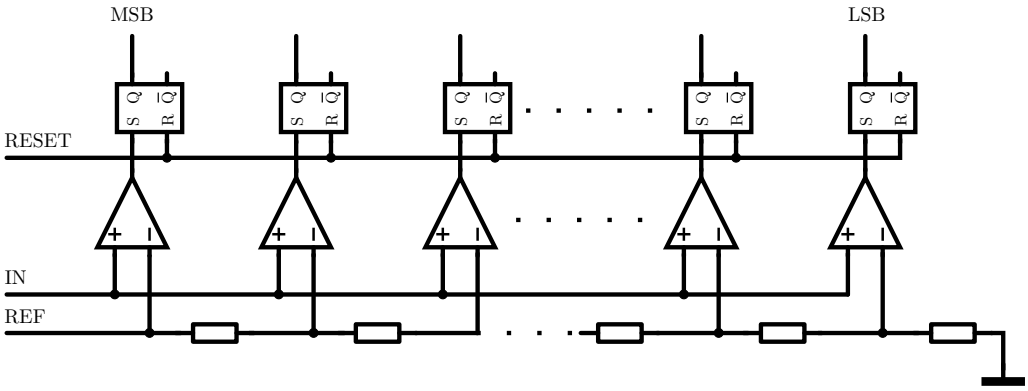


Figure 2.6: Digital PDH

2.2.8 CMOS PDH

The most common peak detector in CMOS technology is the topology presented in [6]. An OTA is used to charge a hold capacitor through a current mirror. The feedback of the circuit is bufferd by a voltage follower. Replacing the usually used diode by a current mirror is the main benefit of this structure. The current mirror brings the same functionality as the diode, current flow only in one direction, but is much easier and smaller to implement in a CMOS process.
When the input voltage increases the OTA sinks current from the rectifying current mirror and the current mirror output starts to charge the hold capacitor until the output voltage is equal to the input voltage. At the moment where the input voltage becomes smaller than the output voltage the OTA would source current and therefore turn-off the rectifying current-mirror. The performance of this circuit mainly depends on the design of the OTA, especially the transconductance and bandwidth and the bandwidth of the voltage follower in the feedback.
One thing that has to be considered during the design, is that if the input differential voltage of the OTA is 0V the output voltage of the OTA would be equal to the supply voltage minus the gate-source voltage of the current-mirror load. This would still result in a current through the rectifying current mirror, hence also with 0V input differential voltage at the OTA the hold capacitor is still charged until the output voltage of the OTA is high enough to fully turn-off the rectifying current mirror. Increasing the voltage gain of the OTA minimizes this pulse-width dependent offset. Especially for PDH circuits used for fast pulses this offset can be very high because high speed OTAs usually suffer low voltage gain.
A detailed analysis of the topology regarding stability, errors and limitation was done in [7] and an improved version was presented in [8].

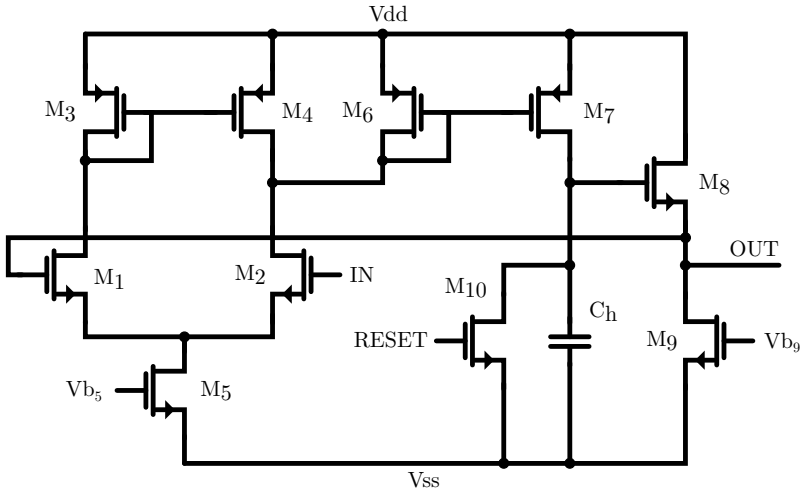


Figure 2.7: CMOS PDH

Chapter 3

PDH Simulation and Test Setup

For generating a simulation and test setup the most important parameters of a peak detect and hold circuit have to be defined. The first of these parameters is the accuracy, it should describe how accurate an input peak voltage is measured. Another parameter is the dynamic range, this range describes the ratio between the minimum and the maximum input voltage which can be measured with acceptable accuracy. One aspect regarding speed is the minimum detectable pulse width, which is again defined by an acceptable error in accuracy for example -3dB. Thinking on the fact that the rise-time of the input signal effects the output voltage this can be treated as a parameter rise-time dependency or can be integrated into accuracy. Summarizing this parameters:

- Accuracy
- Dynamic range
- Minimum detectable pulse-width
- Rise-time dependency
- Hold-time / droop-rate

For a simulation and test setup which is capable of measuring all this parameters a lot of data and data processing is required. To verify these parameters in the whole input voltage voltage range signal pulses with different rise-times and pulse widths have to be applied to the circuit over the whole input voltage range.

3.1 Analysis Types

To test for all this parameters three different analysis were defined which all use a part wise linear (PWL) voltage pulse as input signal. For characterizing the behavior over the whole input voltage range, each pulse shape is applied with different amplitudes to the circuit.

3.1.1 Long-time analysis

During this analysis a pulse with slow rise-time, fall-time and long pulse-width is used as input voltage. This gives information about the basic transfer characteristic of the circuit, without pushing it to its limits. In this analysis also the voltage droop of the held voltage is analyzed by sampling the output voltage after different time steps. The waveform of the pulse in time domain for an amplitude of 1V is shown in the figure below, with a rise- and fall-time of 4ns and a pulse-width of 10ns.

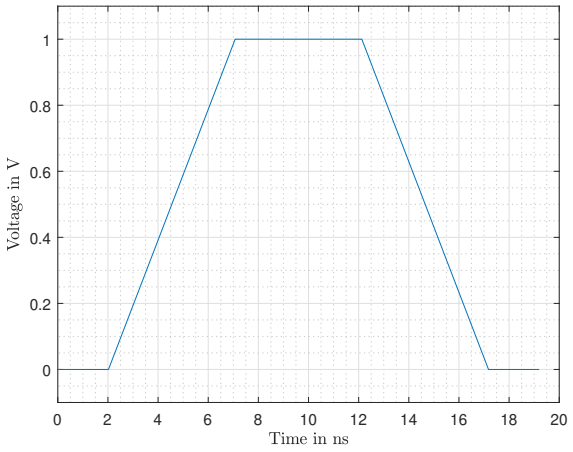


Figure 3.1: Waveform for Long-Time Analysis

3.1.2 Pulse-Width Analysis

This analysis uses pulses with different pulse-widths but constant rise- and fall-times. Testing with different pulse shapes requires a lot of time because for each waveform the amplitude has to be varied from very small voltages to the maximum voltage.
By using very narrow pulses information about the speed of the circuit should be generated, the minimum detectable pulse-width can be determined by comparing the result of long pulses with very short pulses. For the reason that also pulses with very small pulse width should be used the rise- and fall-time has to be fast during this analysis. For all pulses the rise- and fall-time is 800ps and the pulse-width is varied, all pulses are shown in figure 3.2 for an amplitude of 1V.

3.1.3 Rise-Time Analysis

To obtain information on rise-time dependence a long pulse is used with different rise-times but constant slow fall-time. The reason for using a long pulse is that this analysis should only provide information about rise-time dependence, but not about the effects of pulse width.

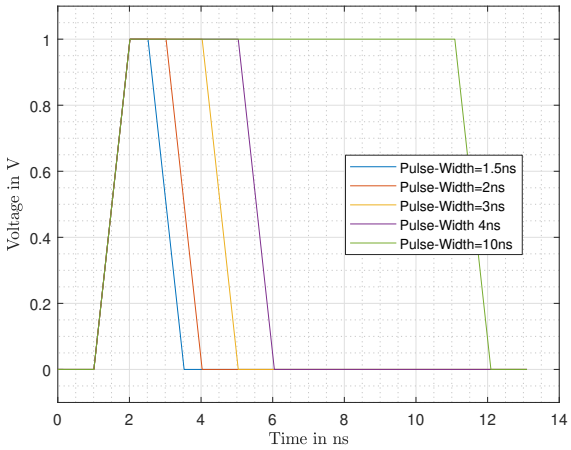


Figure 3.2: Waveform for Pulse-width Analysis

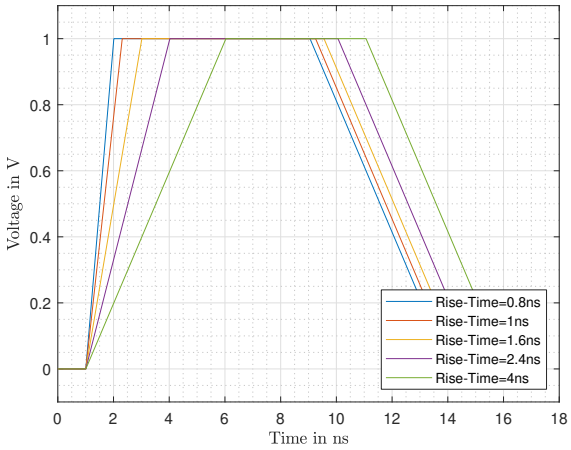


Figure 3.3: Waveform for Rise-Time Analysis

3.2 Ideal Characteristic and Data Visualization

For understanding the simulation and evaluation results an appropriate way is required for interpreting the data. The easiest and probably also the most understandable visualization is a plot of the output voltage vs the input peak voltage. Such a plot for the ideal characteristic of a peak detect and hold is shown in figure 3.4. Plotting data-sets of the response to pulses with different pulse-widths shows how the transfer characteristic changes with pulse-width, the same can be done for different rise-times.
For better insight also a plot of the error of the output voltage referred to the input peak voltage in percent vs input voltage is very useful. Detailed analysis of this data also enables the correction for constant offset or overshoot voltages in the followed data processing.

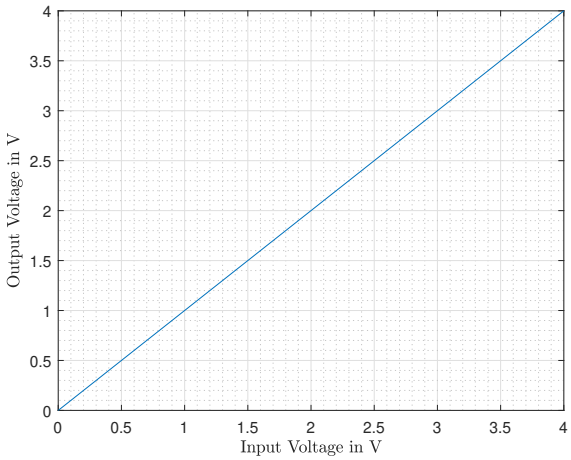


Figure 3.4: PDH Ideal Transfer-Characteristic

3.3 Simulation Setup

A simulation setup which is able to compute the in section 3.1 analysis types was created using the free SPICE based software LT-SPICE from Analog Devices. The setup only consists of a PWL-voltage source which is used for generating the input voltage pulses and spice-directives.
In figure 3.5 the whole setup is shown with all required SPICE-Directives, for performing a certain analysis all commands in the associated box have to be applied, the commands in the other boxes must be indicated as comment by adding a semicolon at the beginning. The circuit only consists of a PWL-voltage source which is connected to the PDH which can be included as sub-circuit or directly into the setup and the output of the PDH has to be labeled as ”out”.
The result of the simulated analysis can be taken from the simulation log-file and further processed using MATLAB to generate the plots described in section 3.2.

3.3.1 Simulation Setup Description

For the rise-time and pulse-width analysis a transient simulation is performed for 101ns with maximum time step of 10ps, for the long-time analysis a simulation time of 5.1us is used.

Because the the rise-time is defined as the time difference between 10% and 90% of the pulse amplitude at the rising edge, the rise-times must be converted so they can be applied to the PWL-voltage source. Also the fall-time and the pulse-widths needs to be converted to fit the definitions for the PWL source, the used equations are listed below.

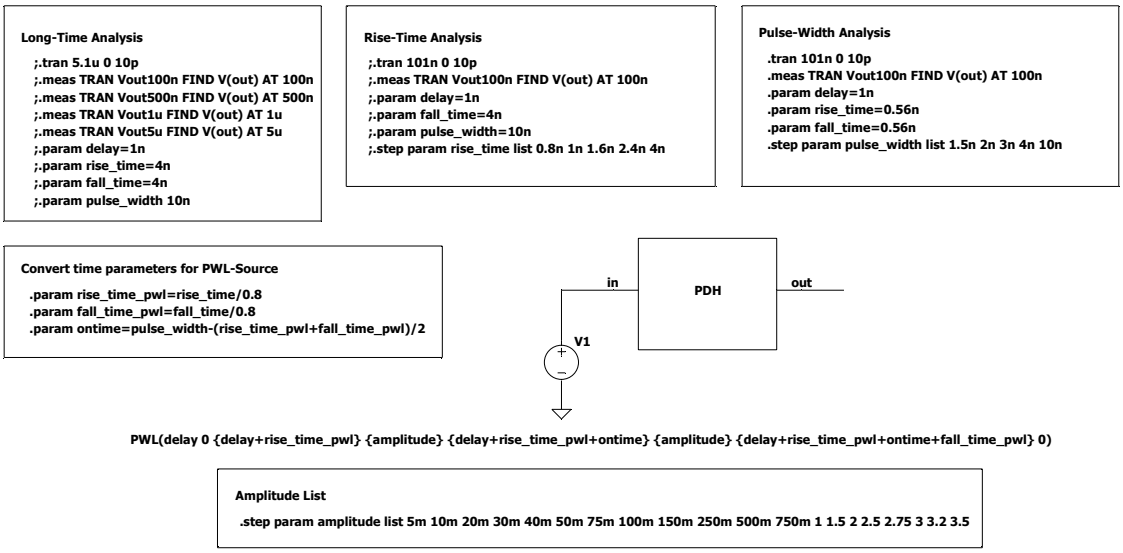


Figure 3.5: LT-SPICE Simulation Setup

$$t_{rPWL} = \frac{t_r}{0.8} \tag{3.1}$$

$$t_{fPWL} = \frac{t_f}{0.8} \tag{3.2}$$

$$PW_{PWL} = PW - \frac{(t_{rPWL} + t_{fPWL})}{2} \tag{3.3}$$

The conversion of the time parameters is done using .param commands with the described equations. For the variation of the amplitude a list with values for the amplitude is also provided as .param command. With the .meas SPICE-Directive the output voltage is measured after a certain simulation time, the result of this measurement is stored in the simulation log-file.

3.4 Automated Test Setup

Creating a test setup which is able to generate this broad range of pulses and measure the output voltage of the PDH circuit after a specific time requires a complex measurement setup with high degree of automation.
For the generation of different waveforms an arbitrary waveform generator (AWG) with sampling rate up to 10GS/s is used, which provides very good adaptability even for narrow pulse shapes. Because the output voltage of the used AWG is limited to 500mV an amplifier has to be integrated into the test system which is able to provide voltages up to at least 3.5V. For changing the pulse amplitude two step-attenuators with up to 47dB overall attenuation are used combined with the ability of reducing the voltage of the AWG by 6dB. This results in a dynamic range of 53dB for the test setup, enabling that tests can

be performed across the whole input voltage range of the PDH. The measurement of the PDH output and input voltage is done by an oscilloscope with 4GHz bandwidth using a 50Ω input.
For circuits where certain parameters as bias current or bias voltages can be changed, this is done by general purpose IO-cards which offer some DACs and ADCs.
The whole analysis is controlled by a python program on a control-PC via GPIB over Ethernet, serial communication and USB. A block diagram is provided in figure 3.6.

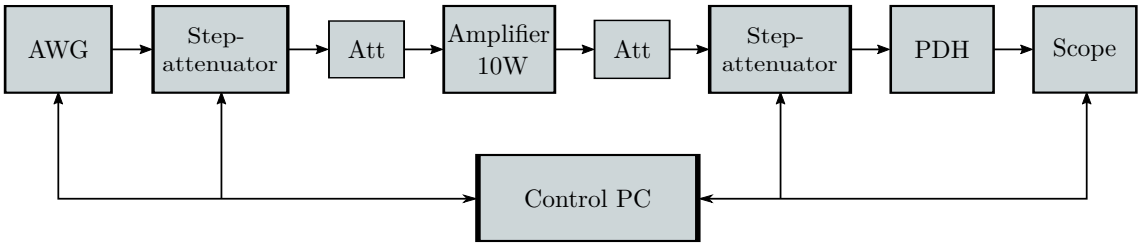


Figure 3.6: Test Setup Block Diagram

3.4.1 Instruments

To generate this fast pulses high sophisticated test equipment is required, the most important instruments of the setup are listed here:

- Tektronix AWG7102 Arbitrary Waveform Generator
- Bonn Elektronik amp **NUMBER**, Amplifier Research 10W1000AM3
- Rohde&Schwarz RTO2044
- NI USB-6001

The AWG features a fast sampling rate of 10GS/s which gives 100ps time resolution, but the amplitude should not be reduced to very small voltages because of limited signal to noise ratio (SNR). For this reason the software controlled step-attenuators are included for adjusting the amplitude. Using GPIB over Ethernet the desired waveform is transferred to the AWG and the settings as sampling-rate and amplitude can be adjusted.

With 30W and a frequency range from 30MHz to 1000MHz the amplifier from BONN Elektronik can generate the required output voltages while providing fast rise-times. This frequency range brings a problem for longer pulses because the low frequency content of the signal is attenuated by a high-pass behavior. A longer PWL-pulse would have a voltage decrease over time and providing a constant amplitude is not possible by applying the desired pulse at the input. This problem can be bypassed by adjusting the input pulse shape of the amplifier for the frequency behavior of the amplifier, the used principle is explained in the section 3.4.2.
The amplifier from Amplifier Reasearch supports a slightly slower rise-time but for that it has a frequency range starting from 500kHz up to 1GHz eliminating the need for pulse-predistortion. The second advantage of this amplifier is that the overshoot at fast rise-times is lower allowing more accurate measurements.

3.4.2 Pulse Predistortion

Adding pulse-pedistortion solves the problem of attenuated low-frequency content. Pulse predistortion means that the input signal of the amplifier is shaped in a way, that the frequency characteristic of the amplifier is compensated.
To compensate the high-pass frequency characteristic of the amplifier a digital high-pass filter is designed which provides the same impulse response as the amplifier. This is achieved by measuring the impulse response of the amplifier and than adjusting the digital filter to match this response. When the designed digital high-pass filter is inverted by just interchanging the numerator- and denominator-coefficients of the filter, this filter is able to cancel the frequency characteristic of the amplifier. This inverse high-pass filter is applied to the desired PWL-voltage and then transferred to the AWG, the output of the amplifier is now able to provide pulses with constant amplitude.

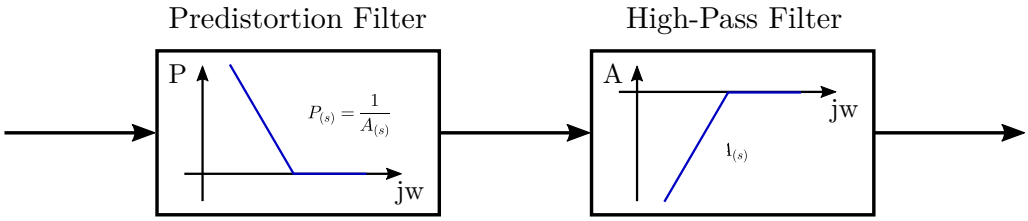


Figure 3.7: Predistortion Principle for High-Pass Filter

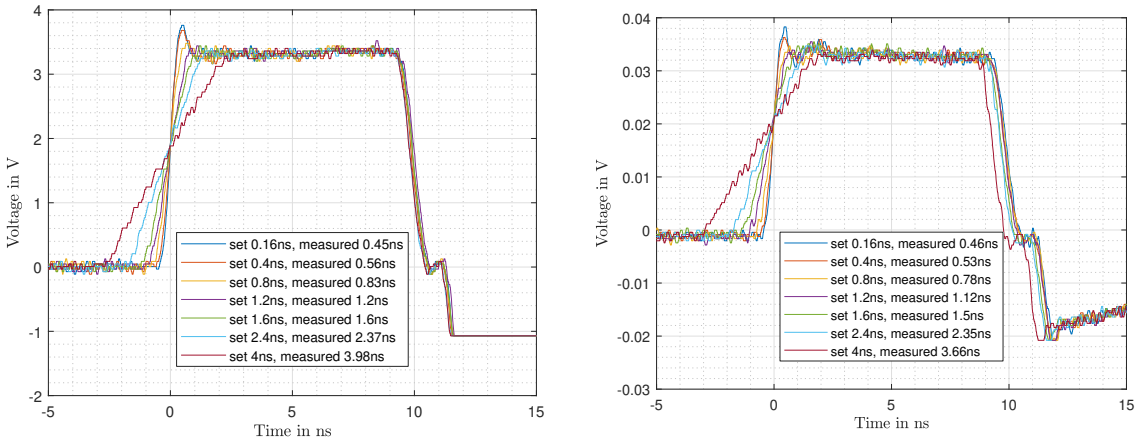
Due to this inverse digital high-pass filter the output voltage of the AWG is continuously increased by a certain amount of voltage to cancel the low-frequency attenuation. Because of this technique the amplitude of the AWG needs some margin to the maximum possible amplitude of the AWG, therefore the amplitude of the AWG was set to half of the maximum. Using this settings pulses up to 20ns length can be generated without much voltage decrease over time. This amplitude still allows to reduce the amplitude of the AWG by 6dB for generating small pulses.

```
1 def pulse_predistortion_inv_hp(pulse, fs):
2     # Highpass filter coefficients
3     b, a = signal.butter(1, 8e6, 'high', fs=fs)
4     # Apply inverse highpass filter
5     dist_pulse = signal.lfilter(a, b, pulse)
6     return dist_pulse
```

3.4.3 Measurement of Test Pulses

To verify the generated test pulses, the response of the BONN and the AR amplifier was measured for pulses with 10ns pulse-width and different rise-times. For the BONN amplifier the already discussed predistortion was used to hold the amplitude constant. In figure 3.8 the output voltage of the BONN amplifier is shown for an amplitude of 3.3V and 33mV and in figure 3.9 for the AR amplifier.
The plot legend shows the rise-time which is set by software and the real measured rise-

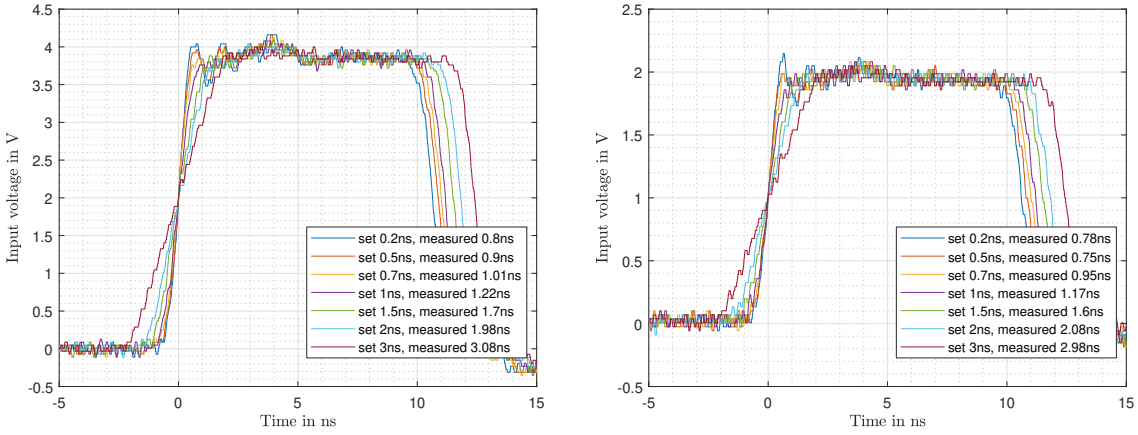
time calculated using MATLAB. From the plot one can obtain that the fastest possible rise-time with the BONN amplifier is 450ps when the rise-time in software is set to 160ps. Meaning that for the rise-times of 450ps and 560ps the value set in software differs from the real value, for all other required rise-times the value in software quite similar. The fastest rise-time of the AR amplifier is 800ps, here the values set in software are different to the real value up to 1.6ns.



(a) 3.3V Amplitude

(b) 33mV Amplitude

Figure 3.8: Test Pulse Measurement for BONN Amplifier



(a) 4V Amplitude

(b) 2V Amplitude

Figure 3.9: Test Pulse Measurement for AR-Amplifier

Pulse Overshoot

In the measurement of the BONN amplifier a rise-time and amplitude dependent overshoot is obtained, this overshoot brings a problem for the measurement of the input peak voltage. Because the input peak voltage is measured with a mathematical max function the overshoot for each rise-time is measured to enable a correction in the data processing. The measured overshoot is provided in table 3.1.

	Overshoot	
Rise-Time	Amplitude=3.3V	Amplitude=1.7V
450ps	12%	9.6%
560ps	10.2%	5.4%
800ps	4.4%	1.1%
1.2ns	3.5%	3.6%
1.6ns	3%	2.2%
2.4ns	2.6%	4.7%
4ns	1.2%	0.1%

Table 3.1: BONN Amplifier Test Pulse Overshoot

With this overshoot value of up to 12% the amplifier is not suitable for accurate measurements, therefore for accurate measurements the amplifier from Amplifier Research is preferred.

3.4.4 Signal Processing

The measurement setup should measure the input peak voltage directly at the input of the PDH and the output voltage. Because some PDH circuits have a constant offset voltage at the output and the input also this voltages have to be measured and be corrected later in the signal processing. The input peak voltage is measured with the max function in python and the output voltage is measured after a defined time after the input voltage triggered the scope. As figure 3.8a and 3.8b show the test setup suffers from broad band noise from the amplifier. Due to this noise, taking only one sample of the voltage is very inaccurate, to prevent this inaccuracy the measured signals are appropriately filtered in the software. The used filters are described in the following sections.

Filter for PDH Output Voltage and Offset Voltages

A filter is applied to the output and input voltage to enable the measurement of offset voltage before the input pulse arrives and for the measurement of the output voltage after peak detection. For both measurements the transient behavior of the filter is not important because the offset voltage is constant and also the output voltage should be quite constant in hold mode. This simple demands on the filter allow the use of a moving average filter which is implemented in python using the convolution function from the numpy library. The averaging filter uses an averaging window of 6ns, the result is shown on an exemplary PDH output voltage in figure 3.10.

```
1       #Query scope resolution
2       scope_res = float(rto.query_str("ACQ:RES?"))
3       #Define averaging-window size in samples
4       window = 6e-9   #6ns averaging window
5       window_samples = int(window/scope_res)
6       #Apply averaging filter to data by using convolution
```

```
7       data_ch1_av_filt = np.convolve(data_ch1, np.ones(window_samples) /
        ↪  window_samples, mode='same')
8       data_ch2_av_filt = np.convolve(data_ch2, np.ones(window_samples) /
        ↪  window_samples, mode='same')
```

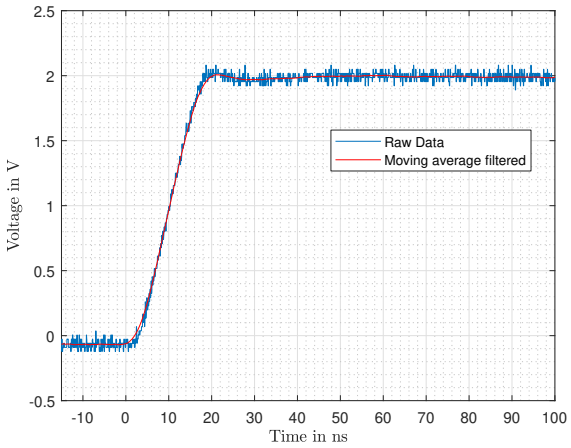


Figure 3.10: Moving-Average Filtered Signal

Filter for PDH Input Peak Voltage

Filtering the measured input voltage requires a filter which also allows fast transient signals because also very narrow pulses are used. A third order butterworth low-pass filter with a cut-off frequency of 1GHz was designed using the python signal library. The filter was applied to the signal using the filtfilt function which applies the filter twice to the signal while providing zero phase shift. In figure 3.11 a filtered and a raw pulse are shown.

```
1       #Query scope resolution
2       scope_res = float(rto.query_str("ACQ:RES?"))
3       #Low-pass filter coefficients
4       b, a = signal.butter(3, 1e9, 'low', fs=1/scope_res)
5       #Apply filter to data
6       ch2_butter_filt = signal.filtfilt(b, a, data_ch2)
```

3.4.5 PDH Operating Point Adjustment

Using general purpose IO-cards which are connected via USB to the computer, bias voltages and bias currents are measured with the ADC and adjusted using the DAC. A simple regulation with a while loop was implemented which increases the DAC output voltage when the measured value is to low until it is in a certain ϵ-region. For preventing an infinite loop a maximum number of iterations is defined which prints an error when reached.

3.4.6 ATE Software

The software for the automated test setup was created using Python. This programming language was chosen because of its advanced capabilities in interfacing with measurement

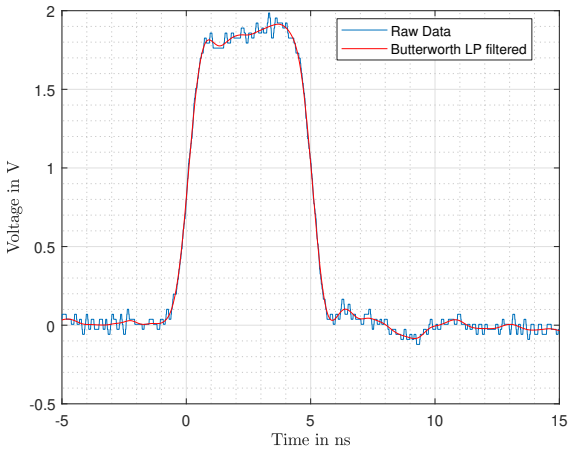


Figure 3.11: Butterworth Low-Pass Filtered Signal

systems and very well data processing options.
Executing the software automatically starts all the analysis discussed in section 3.1 and automatically generates the required plots to gain information about the circuit performance. For further data analysis the measurement results are saved as csv file.

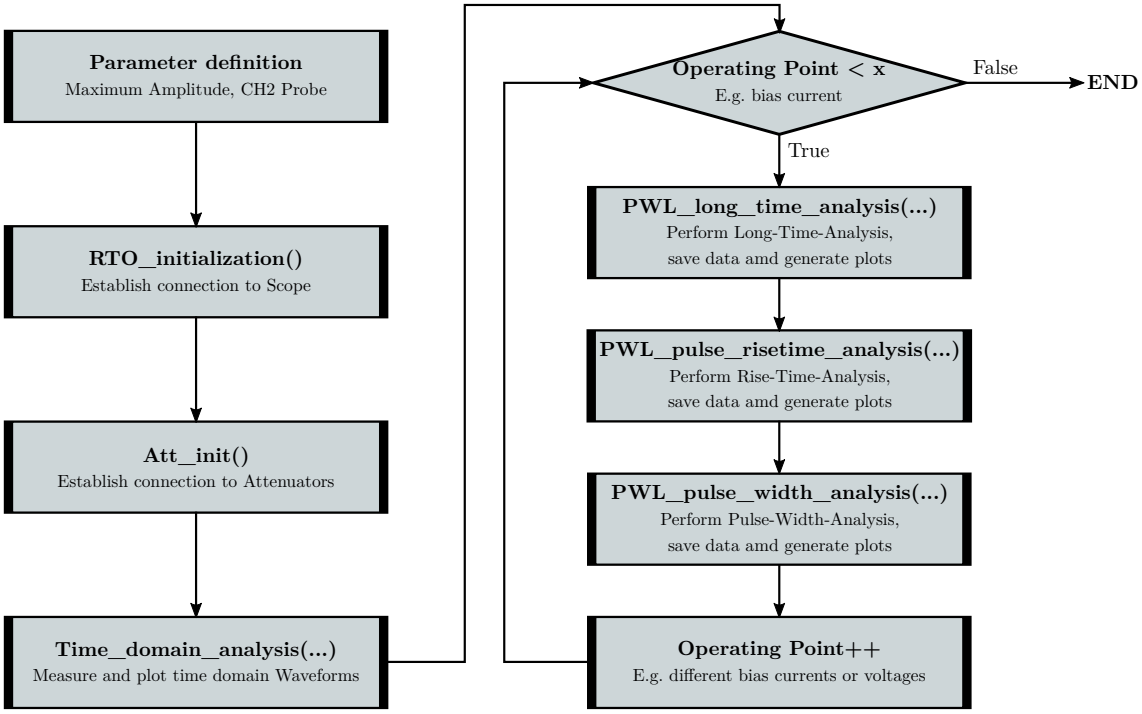


Figure 3.12: ATE Software Flow-Chart

The first part in the software is to inform the user about what is happening, this is done with the message "Operation started, you can go for a coffee, this may take a while!", after this the the real operation starts.

At the beginning information about the measurement setup is defined such as the maximum pulse amplitude at the circuit input with all programmable attenuators at 0dB. Also the value of the probing resistor which is used to measure the input voltage has to be set so that a probe factor can be calculated (using the 50Ω input of the oscilloscope). The magnitude which should be used by the AWG is defined to -0.25, because an inverting amplifier is used the amplitude has to be negative to get a positive pulse to the circuit. As value for the maximum attenuation 53 is set, this results from the combination of the 15dB and 32dB programmable attenuator together with reducing the AWG amplitude by 6dB.

In the next step the connection to the oscilloscope is established and the device is initialized to:

- Horizontal offset to 10% of the screen
- CH1 and CH2 on
- CH1 and CH2 coupling to DC 50Ω
- Trigger mode ”normal”
- Trigger on positive edge of CH2

Then the connection to the attenuators is initialized, after this the functions for the different analysis types start. For generating some time domain plots of the PDH response a function is called which performes a time domain analysis. this is followed by the functions for Long-Time analysis, Rise-Time analysis and Pulse-Width analysis.

Analysis Implementation

All implemented functions for the different analysis types are very similar in their principle. The required waveform data is generated and transferred to the AWG. In the next step the attenuation is varied using a for-loop.

In figure 3.13 the program flow is shown for the function "PWL_long_time_analysis(...)", in the function of the rise-time and pulse-width analysis the loop for changing the attenuation is included in a second loop for varying the pulse shape. In the function "RTO_save_CHs_longtime(...)" the input peak voltage and output voltage is stored. The results of this function are already corrected for offset voltages. For better measurement results the filters described in section 3.4.4 are used.

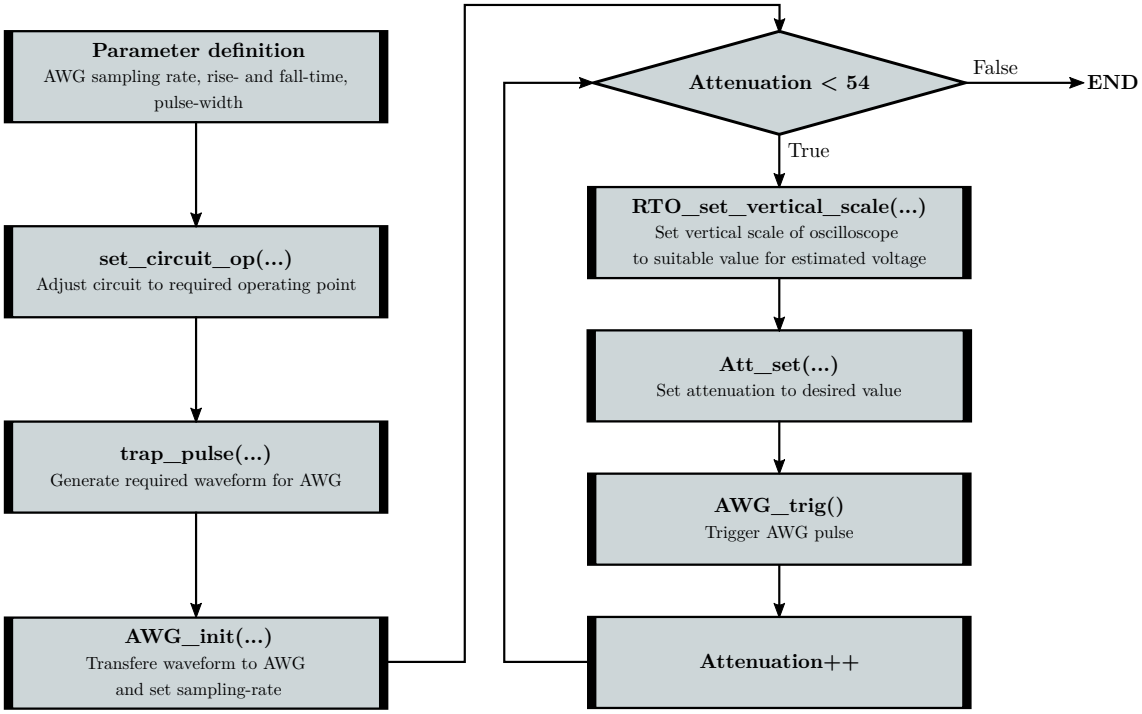


Figure 3.13: Long-Time Analysis Flow-Chart

Chapter 4

PDH Using Bipolar Transistors

This section presents a novel peak detect and hold (PDH) circuit which is intended to be used for measuring the peak voltage of electromagnetic-field probes which capture the fields generated by electrostatic discharge (ESD) events in non-grounded portable devices. A topology using discrete bipolar transistors in a structure with OTA and common-base configuration with software adjustable bias point turned out to satisfy the requirements best. The software adjustable operating point thereby allows maximum power efficiency by matching the circuit performance to the sensor response. Circuit design using circuit simulation is described and potential issues are discussed. The final circuit offers PDH operation for pulse-widths down to 1.5ns by supporting a dynamic range 26.7dB with a power consumption of 153mW and 37dB for a pulse-width of 3ns with only 108mW of power consumption. With an accuracy of 10% which is acceptable for ESD measurements and a very long hold time of up to 10us by a voltage droop of 10%.

4.1 Introduction

The measurement of very fast pulses in the lower nanosecond range is very challenging. Often these pulses are measured using digital storage oscilloscopes with suitable sampling-rate and bandwidth, this option is very suitable for laboratory measurements which are done manually. In systems where the peak-voltage of pulses should be measured, stored and maybe further processed automatically, the measurement of wave-forms in this time range is usually done with PDH-circuits. These circuits are very cost effective solutions which capture a peak voltage and hold it for a certain amount of time.
Typical applications which require the use of PDH-circuits are measurement systems where a sensor produces a very narrow pulse, for example in spectrometers which use semiconductor-detectors and the information is stored in the peak value. Many papers have been published for these applications of mass spectrometers, time-of-flight spectrometers, magnetic spectrometers and optical absorption spectrometers. In these applications the output of the detector is a train of narrow current pulses which is converted to a voltage signal by a charge sensitive amplifier (CSA) or transimpedance amplifier (TIA) and fed to a PDH which enables analog to digital conversion with low sampling-rate.
The most important parameters of these circuits are:

- Amplitude accuracy
- Dependency on pulse shape, minimum detectable pulse-width
- Dynamic range

Side parameters as power consumption also should be considered as those often inhibit battery powered operation. Some of these parameters of course have certain correlation to each other.

4.2 Peak Detect and Hold Topologies

An overview on existing analog and digital PDH circuits for nanosecond pulses, by focusing on digital circuits and their effective trigger signal was already reported [4]. For analog peak detect and hold circuits there are three basic topologies:

- Follow and hold circuit which is triggered by a peak detection trigger signal to change to hold mode
- A triggered follow and hold circuit combined with a diode capacitor configuration
- A diode capacitor configuration in combination with an amplifier
- PNP common-base capacitor configuration with amplifier
- OTA with current-mirror

4.2.1 Triggered Follow and Hold

Circuits using follow and hold structures require a trigger signal which changes the operation from follow- to hold-mode when the input peak voltage is reached. For fast pulses the generation of this can be very challenging, because the timing of the trigger event has the highest impact on the performance of the circuit. Most common methods for the generation of these trigger signals are leading edge discrimination (LED) and constant fraction discrimination (CFD), whereas the first strongly depends on the pulse shape [9].

4.2.2 Triggered Follow and Hold with Diode Capacitor Configuration

The concept of using a SOTA in combination with a diode capacitor combination reduces the effect of trigger timing because the charging current of the hold capacitor is rectified by the diode. The option for hold mode thereby narrows down the reverse leakage current through the diode in hold mode. By the use of a diode the disadvantages of its nonlinear behavior and dynamic on-resistance are introduced [10].

4.2.3 Diode Capacitor Configuration with Amplifier

At a diode capacitor configuration with amplifier the amplifier is in voltage follower configuration and the diode rectifies the charging current. By supplying the amplifier feedback voltage with the voltage at the hold capacitor the diode drop is eliminated. In case of amplifiers with low input impedance a voltage buffer is used to provide the feedback voltage.

These topology is very common because of simplicity and good performance. The disadvantages are the nonlinear behavior of the diode, leakage current and cross-talk through the diodes parasitic capacitance [11][12][13].

4.2.4 Common-Base Configuration with Amplifier

The use of a pnp-transistor in common base configuration with hold capacitor supported by an amplifier offers many advantages compared to the use of a diode. The main advantage is the current gain which reduces the requirement of voltage swing of the amplifier output and the adjustable bias voltage at the base which enables a suitable adjustment of the operating point of the amplifier. Drawbacks as nonlinear behavior, reverse leakage current and cross-talk through the Base-Emitter capacitance still exist [14][15][16].

4.2.5 OTA with Current-Mirror

In integrated circuit technology it is usually preferred to use a current mirror for rectification instead of a diode [6] [7] [8] [17] [18].

4.3 Proposed Circuit

The concept of the proposed circuit is shown in figure 4.1 the full schematic without dicharge switch (JFET) in figure 4.2 and a schematic including all component values is provided in Appendix A. the circuit consists of an OTA with clamping network at the output, followed by a pnp-transistor in common-base configuration. This transistor only allows charging the hold capacitor by providing low input impedance to the OTA output and blocks the discharge of the hold capacitor by a very high output impedance. The voltage used for the feedback of the OTA can not be directly supplied by the node of the hold capacitor because of the high base current of the input differential pair transistor. Therefore a high-speed voltage buffer which provides low input bias current was introduced, consisting of a n-channel JFET and a npn-transistor. The JFET provides low input bias current and transconductance, whereby the npn-transistor provides current gain to drive capacitive loads. For adjusting circuit parameters as tail bias current, feedback bias current and common-base bias voltage the circuit is controlled using DACs and ADCs of a NI USB-60001 general purpose IO-card.
The novelty of the circuit lays in the combination of a high-speed OTA and common-base rectifier together with a high-speed voltage buffer while providing full adaptability by software. By software adjustment the dynamic range / speed of the circuit can be adjusted for the required operation. Especially in multi-channel PDH-Systems this enables matching the requirements of each channel to the expected signal. Thinking of a PDH-system which is connected to different electro-magnetic field sensors, where some of the sensors generate slow pulses, whereas others produces fast signals matching the applied PDH-circuit to the requirement of the sensor allows maximum power efficiency.

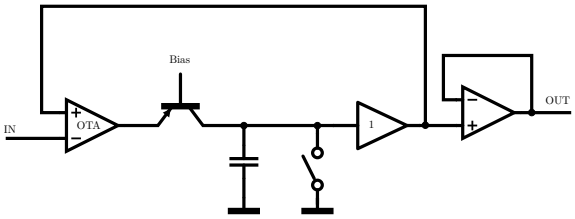


Figure 4.1: Proposed Circuit Concept

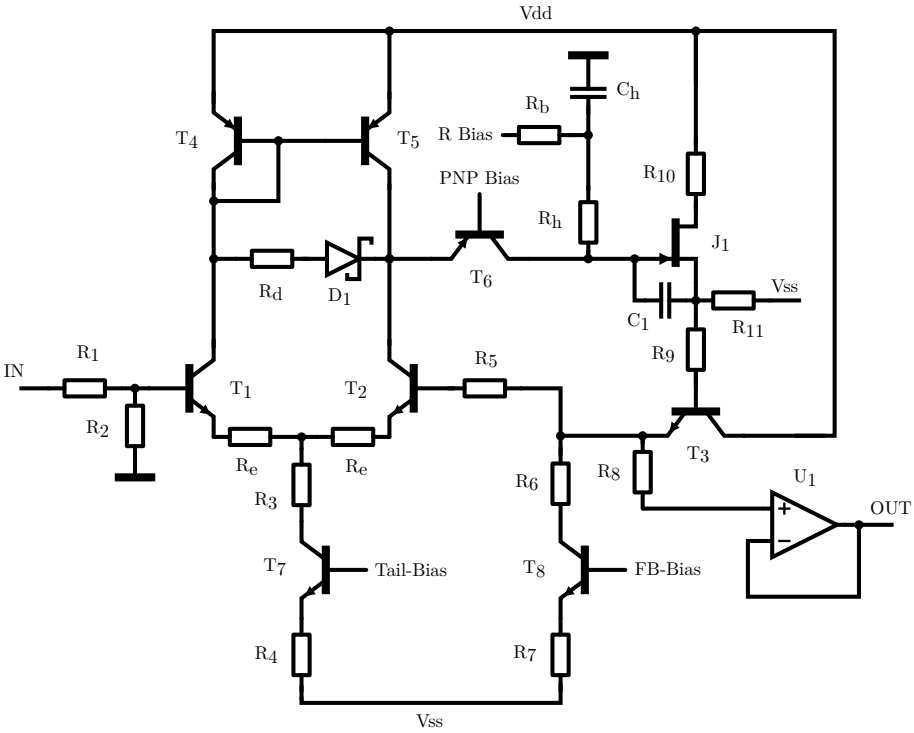


Figure 4.2: Proposed Circuit Schematic

4.3.1 OTA

For the OTA the very basic structure of an input differential pair with pnp-current-mirror load was chosen, biased by a tail current source. The transistors of the input differential pair use a small value emitter degeneration resistor. These resistors provide a certain amount of negative feedback and extend the amplifiers linear operating region to approximately $R_e \cdot I_e$ [19]. This extension of the linear region is especially important to reduce rise-time dependency. Thereby the internal rise-time of the voltage across the hold capacitor is reduced making delay of the feedback less critical. The clamping of the output voltage is done by a schottky diode with a series resistor connected between diode connected pnp-transistor of the current mirror load and the output of the amplifier. When the circuit is in peak detection mode this diode is reverse biased, after a peak is detected and the circuit changes to hold mode, the clamping network starts conducting and supplies half of the tail current, resulting in a minimum output voltage of:

$$V_{OTAmin} = V_{DD} - V_{EB} - V_{clamp} \tag{4.1}$$

where V_{EB} is the Emitter-Base voltage of the pnp-current-mirror load and V_{clamp} is the voltage drop of the clamping network, with:

$$V_{clamp} = \frac{I_{tail}}{2} R_d + V_{diode} \tag{4.2}$$

This clamping of the output voltage reduces the effect of the cross-talk through the rectifying pnp-transistors collector-emitter capacitance. The error generated by this cross-talk is defined by:

$$V_{error} = \Delta V_{OTA} \frac{C_{CE}}{C_{CE} + C_{hold}} \tag{4.3}$$

Where ΔV_{OTA} is the voltage swing at the emitter of the rectifying pnp-transistor.

The diode series resistor in the clamping network is required for damping oscillation which can be caused by the parasitic inductance of the clamping diode. When a pulse was detected and the circuit changes its operation to hold mode, the current through the clamping network rises very fast. The node at the cathode of the diode is a high ohmic node in the hold mode because it is connected to the collector of T_5 and T_2 and T_6 is reverse biased. This in combination with the fast current rise through the diode can cause oscillations since the parasitic inductance of the diode and the capacitance at the high ohmic node can form an LC-circuit with positive feedback through the pnp-current mirror. For damping this LC-circuit the series resistor was introduced.

4.3.2 Common-Base Rectifier

This transistor in common-base configuration is used for rectification. Because of the low input impedance provided by the current gain the required voltage swing of the amplifier is reduced. The most important parameters which effect the operation of the circuit are the current gain and the parasitic capacitances. The Emitter-Base capacitance leads to a small amount of negative feedback because when the emitter voltage increases the base voltage also increases, whereas the collector-emitter capacitance discharges the hold capacitor when the emitter voltage decreases and therefore generates an voltage error as described above. By adjusting the bias voltage at the base the output voltage of the OTA can be changed and set to a value so that the clamping network is reverse biased enabling the best performance for the amplifier.

4.3.3 Voltage Buffer

The voltage buffer dose not exactly buffer the voltage of the hold capacitor, but this is not needed in this circuit. In the used buffer structure the offset is defined by:

$$V_{Boff} \approx V_{GS} + V_{BE} \tag{4.4}$$

Where V_{GS} is the gate-source voltage of the n-channel JFET and and V_{BE} the base-emitter voltage of the npn-transistor, the voltage drop across the series resistor is neglected. Due to the fact that the JFET has negative gate-source voltage the offset cancels out to a certain amount depending on the used bias currents and components.
Because the feedback of the amplifier is with reference to the output voltage of the buffer this offset does not effect the output voltage of the PDH. It only causes an offset at the hold capacitor which is not seen at the output.
It is very important that the voltage buffer has a flat frequency response in the region of operation, deviations from this flat response result in too large or too low output signals depending on the input pulse frequency content. For achieving this flat frequency response also for higher frequencies a speed up capacitor is added between the gate and drain of the JFET.

4.3.4 Important Design Aspects

Designing a PDH circuit for very narrow time ranges can be very challenging and requires steady nerves to go further. During the design and verification oscillations in GHz range and strong mismatch between simulation and measurement occurred. The most important and critical aspects are listed below:

- Small and compact structures keeping time delay low
- Bias current sources should be decoupled with resistors as close as posible to the rf-part by values as high as possible to avoid rf-stabs
- Minimize parasitic capacitances and inductances
- In simulation consider expected or present feedback delay
- Avoid dual-package components in bias or feedback transistors

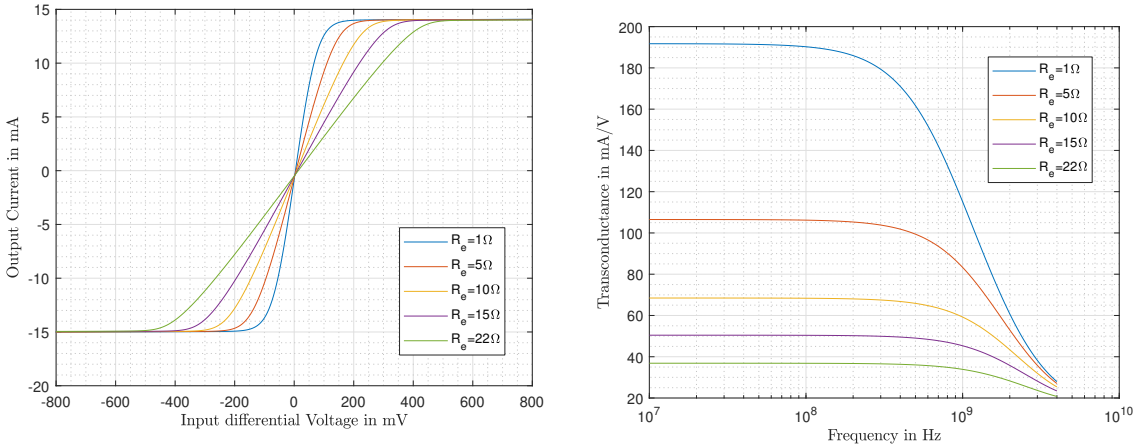
4.4 Circuit Simulation

For the simulation of the circuit the in section 3.3 described test bench in LT-SPICE was used, which should give an overview of the most important specifications of the circuit. The core parts as the OTA and the voltage buffer were tuned individually in simulation for achieving best performance.

4.4.1 OTA Performance Analysis

The performance of the OTA was simulated using a tail bias current of 15mA. For simulating the transfer characteristic the output of the amplifier is connected to a voltage source with a value of 4.26V which is the value the output has in steady state operation. In figure 4.3a the output current versus input differential voltage is shown for different values of the emitter degeneration resistor. It clearly shows the extension of the linear operating

region of the output current for higher resistance values. Whereas figure 4.3b shows the frequency characteristic of the transconductance for different resistor values. Of course this simulation does not include parasitic elements (especially capacitance) of the PCB, the clamping network and the common-base rectifier, so the performance in reality will be slightly worse.



(a) Output Current vs Input Differential Voltage

(b) Transconductance vs Frequency

Figure 4.3: OTA Simulation Results

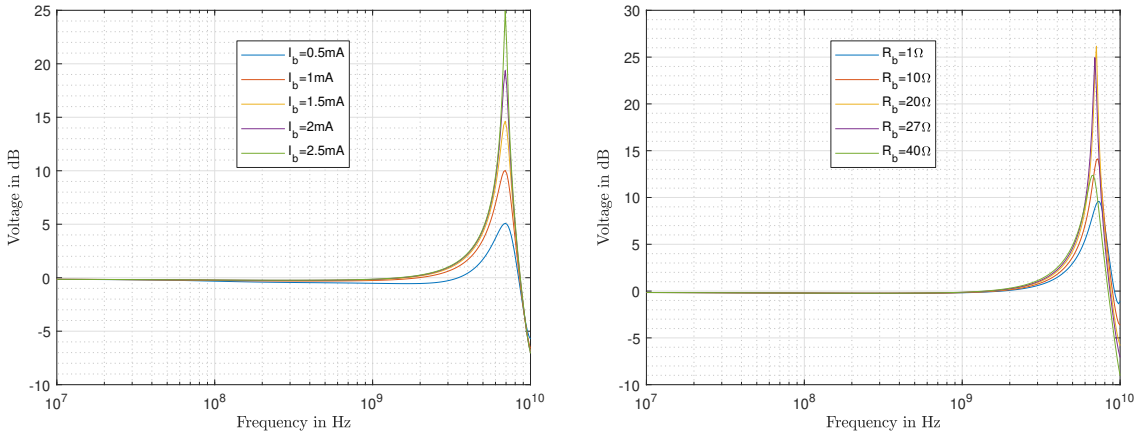
4.4.2 Voltage Buffer Performance Analysis

For the simulation of the buffer performance only the part of the circuit with the buffer and half of the input differential pair was used. The half of the input differential pair is used as load of the circuit, for generating results as close as possible to reality. From the frequency characteristic in 4.4a the behavior for different bias currents can be obtained. It shows that a bias current of 1.5mA is sufficient for geting a flat frequency response beyond 1GHz. Each deviation would generate freqency dependent offset in the output voltage. Also a gain peak is seen at approximetly 7Ghz which increases with bias current. Figure 4.4b shows that the resonant peak is damped by low and high resistance values. From this plot one would obtain that a very small resistance as 1Ω would be a desirable value.

When looking at the result of the transient simulation in figure 4.5 for the different values of base resistors, strong oscillation is seen for low values.

4.4.3 Full Circuit Simulation

The operating point of the circuit was set by a tail current of 15mA and a bias current of 2.5mA at the bias current source in the feedback. To ensure that the diode which is connected to the output of the OTA is reverse biased, the bias voltage of the common base pnp transistor was adjusted in a way that the voltage across the diode is -50mV. For better matching to reality a delay line was introduced in the feedback with 150ps, which was approximated by distance measurement in the pcb layout.



(a) Frequency Characteristic with $R_b = 27\Omega$ (b) Frequency Characteristic with I_b=2.5mA

Figure 4.4: Buffer Simulation Results

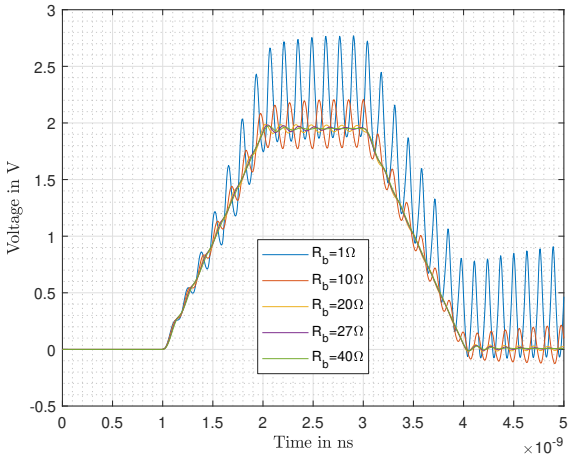


Figure 4.5: Buffer Transient-Simulation

Pulse-Width Analysis

For getting information about the impact of the pulse-width on the output voltage of the circuit a transient simulation was performed with pulses of different width. The output voltage was measured after 100ns and the simulation result is shown in figure 4.7.

Rise-Time Analysis

This analysis should only show the effect of different pulse rise-times on the output voltage, therefore a long pulse with a duration of 10ns and a fall-time of 4ns was used in the transient simulation. As value for the output voltage the voltage was measured after 100ns in the simulation.

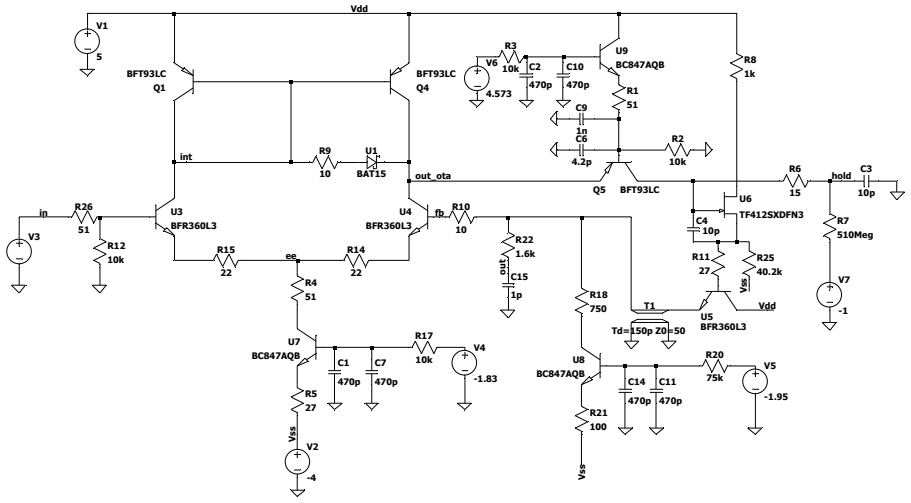
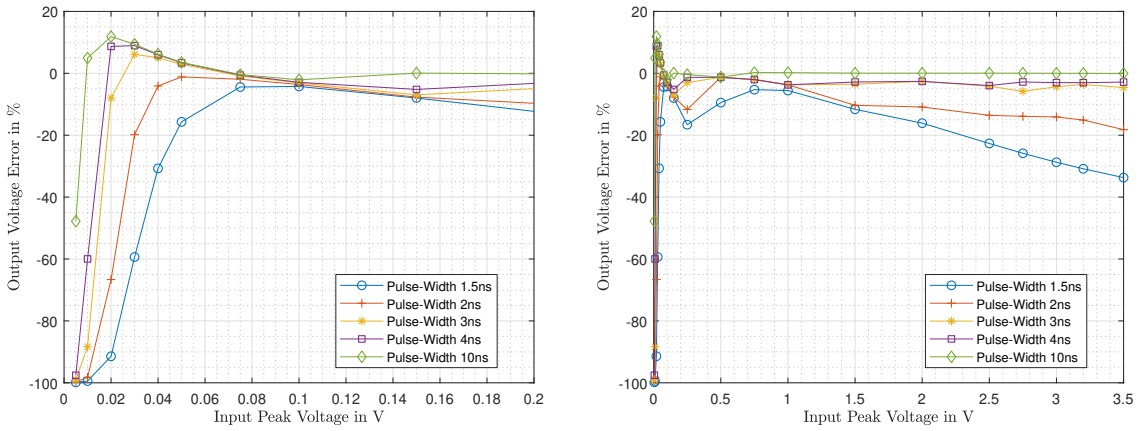


Figure 4.6: PDH Using Bipolar Transistors Simulation Schematic

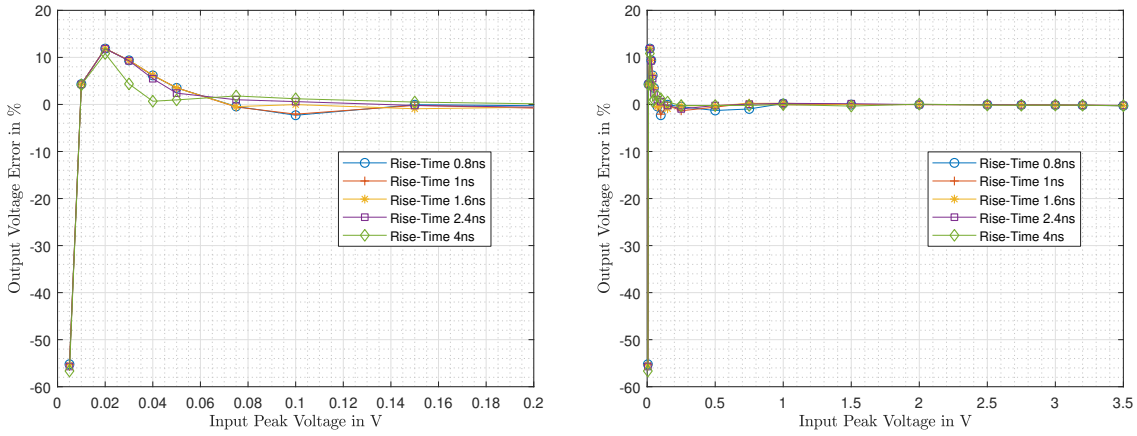


(a) Measurement Error Small-Signals (b) Measurement Error Full Range

Figure 4.7: PDH Pulse-Width Analysis Simulation Voltage Error

4.5 Measurement Results

For verification of the circuit performance a automated test setup was created, which enables the analysis of the circuit response for different voltage-pulses. For the generation of this fast pulses an arbitrary waveform generator (Tektronix AWG7102) with maximum sampling-rate of 10GHz was used. For covering the whole input voltage range a 10W amplifier was used (Amplifier Research 10W1000AM3). The amplitude of the pulse was thereby set by a combination of programmable step attenuators and reduction of the amplitude at the AWG. The output voltage of the circuit was measured after a defined period of time after the input pulse, whereas the input peak voltage was measured with a mathematical max-function using a digital storage oscilloscope.



(a) Voltage Error Small-Signals

(b) Voltage Error Full Range

Figure 4.8: PDH Rise-Time Analysis Simulation Voltage Error

4.5.1 Analysis of Hold-Behavior

In figure 4.9 the measurement result of the long-time analysis is shown by input peak voltage vs output voltage measured after different time periods. As input pulse shape a part-wise-linear (PWL) voltage was used with a rise- and fall-time of 4ns and a pulse width of 10ns. The result shows a very good straight transfer characteristic and shows that the value sampled after 200ns are higher than the input peak voltage but for sampling after $1\mu s$ the result is very close to the ideal value. Also the voltage droop from $1\mu s$ to $5\mu s$ can be obtained as very low. The reason for that is that in the first few 100ns there is a small transient peak which disappears after about 500ns. The reason for that is the speedup capacitor at the voltage buffer which pushes the voltage level to high and than slowly discharges with the bias current of the JFET.

The measurement error after different sampling times shows an overshoot of up to 12% in figure 4.10.

4.5.2 Analysis of Rise-Time Dependency

For analysis of the circuits rise-time dependency the output voltage was sampled after 100ns and a pulse with a pulse-width of 10ns was used with different rise-times. In figure 4.11 the error of the output voltage is shown in the upper and lower voltage range. The result shows a maximum rise-time dependency of 10%, this rise-time dependency is to some extent caused by the used test system, because the amplifier suffers a small amount of overshoot for fast rise-times.

4.5.3 Analysis of Pulse-Width Dependency

The analysis for pulse-width dependency was done using a rise- and fall-time of 0.8ns and the output voltage was again sampled after 100ns. Figure 4.12a shows that the output voltage is lower for smaller pulse-widths in the upper input voltage range. This is a result of limited charging current and limited time for the hold capacitor. A very similar behavior

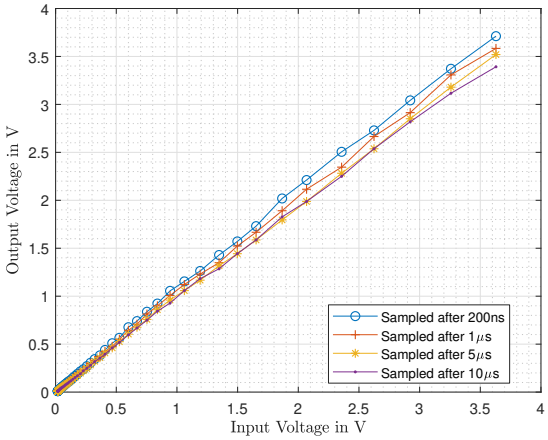


Figure 4.9: PDH Long-Time Analysis Measurement Result

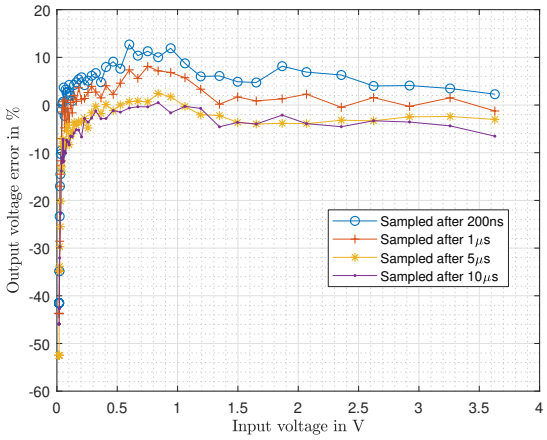
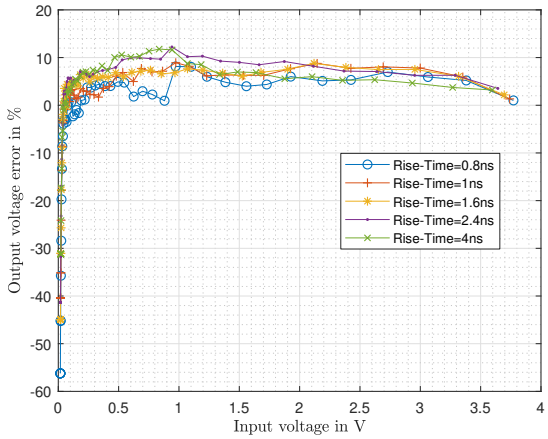
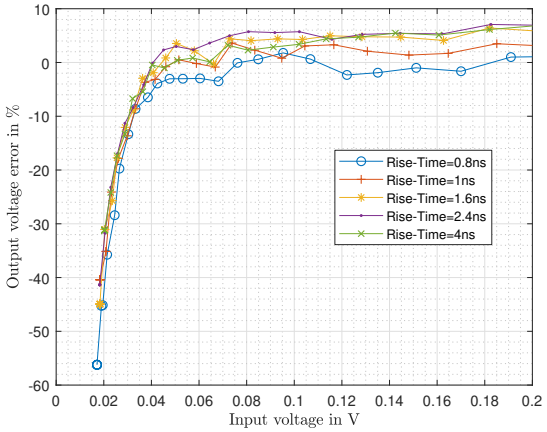


Figure 4.10: PDH Long-Time Analysis Measurement Error



(a) Measurement Error Full Range



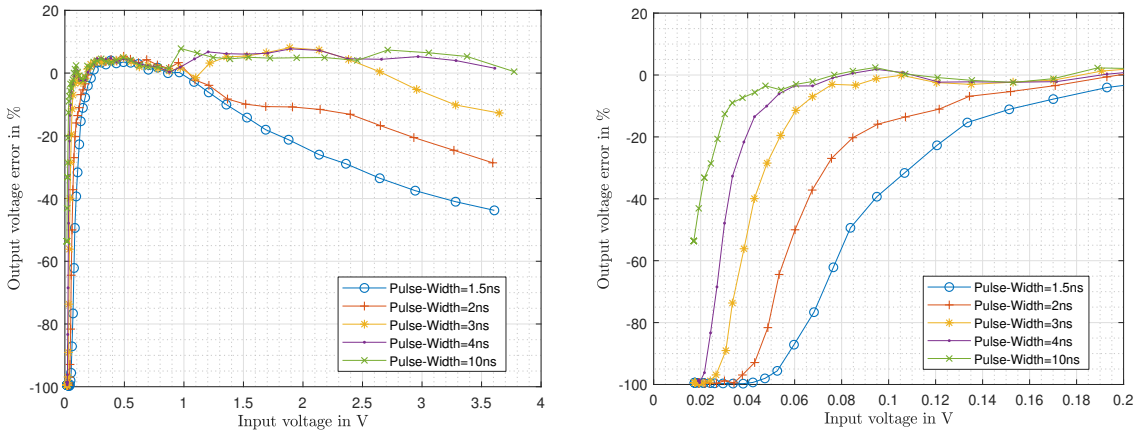
(b) Measurement Error Small-Signals

Figure 4.11: PDH Rise-Time Analysis Measurement Error

can be obtained from figure 4.12b for the lower input voltage range, were the charge current for the hold capacitor is limited because of the limited transconductance of the OTA.

4.5.4 Effect of Tail Bias Current

The most important parameter which is adjustable in the circuit is the tail bias current of the OTA. Changing this parameter highly influences the speed and the maximum charging current for the hold capacitor. In figure 4.13 the output voltage voltage error for different tail bias currents is shown for some narrow pulse widths. First one can see, that the bias current mainly determines the larger voltage region, the smaller voltage region is much less affected, looking at figure 4.13d. It is clearly visible, that a change in the bias current causes the highest variation in the output voltage at the smallest pulse-width of 1.5ns. The larger the pulse-width gets, the less tail bias current is required by the OTA for reaching large output voltages.



(a) Measurement Error Full Range

(b) Measurement Error Small-Signals

Figure 4.12: PDH Pulse-Width Analysis Measurement Error

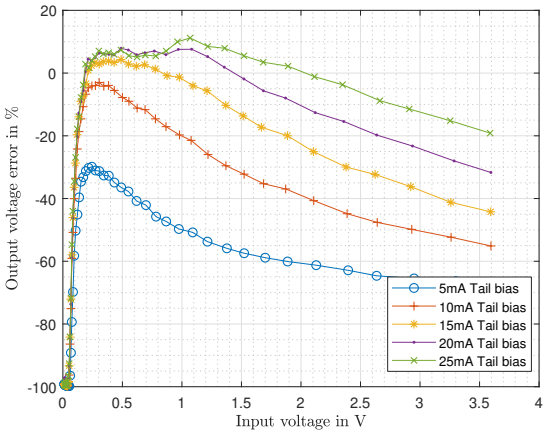
Having this results the necessary tail bias current can be set to a value which fulfills the requirements of dynamic range by using as less bias current as necessary and therefor saving power.

4.5.5 Effect of Feedback Bias Current

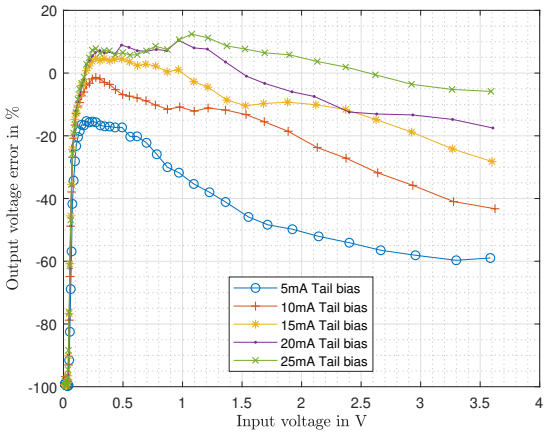
The speed of the voltage buffer used for the feedback of the circuit is mainly determined by its bias current. It is important, that the buffer works up to the highest frequency which occurs in the feedback system, with a gain very close to 0dB. If the buffer is to slow the feedback signal is attenuated at high frequency and the output voltage is to large.
For analysing the required feedback current the circuit was tested with a 10ns wide pulse with a rise- and fall-time of 0.8ns for different bias currents.
Figure 4.14 shows that as expected the higher the tail current the less voltage error. For 0.5mA bias current in the feedback the output voltage is especially at small voltages up to 10% higher and the error for 1mA to 2.5mA is nearly identical. This means a feedback bias current of 1mA already leads to very good results and a higher bias current would only increase power consumption.

4.5.6 Measurement Results Summary

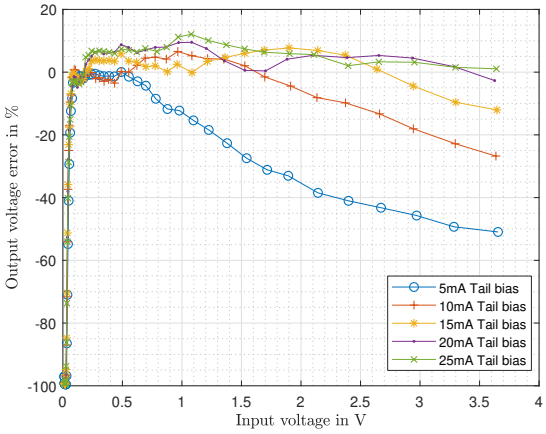
All the measurement results provide a good overview of the important circuit parameters an describe the influence of different bias points very well. For better understanding of the circuit behavior for different pulse-widths and different tail bias currents the results are summarized in table 4.1. For the calculation of the dynamic range a 3db deviation to the ideal output voltage was chosen for the lower- and upper voltage limits. In case there was no 3dB deviation for the higher voltage levels 3.5V was selected as maximum voltage. The circuit should be able to be operated with voltages up to 4V but this was not tested, because the scope of this work focuses on very fast pulses and lower power consumption. For pulse-widths large than 2ns the measurement only up to 3.5V therefore shows a lower dynamic range than possible with this circuit.



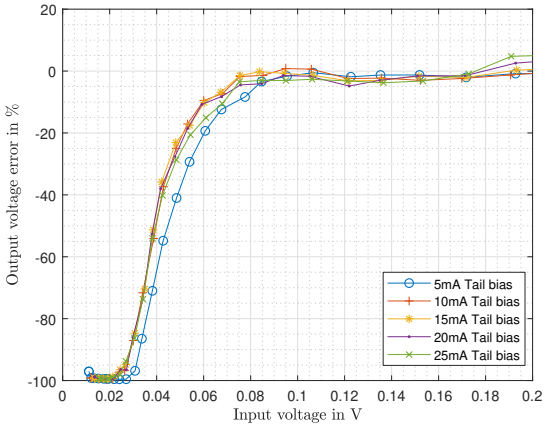
(a) 1.5ns Pulse-Width



(b) 2ns Pulse-Width

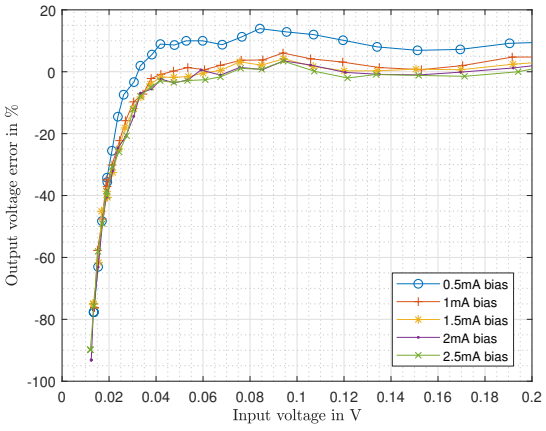


(c) 3ns Pulse-Width

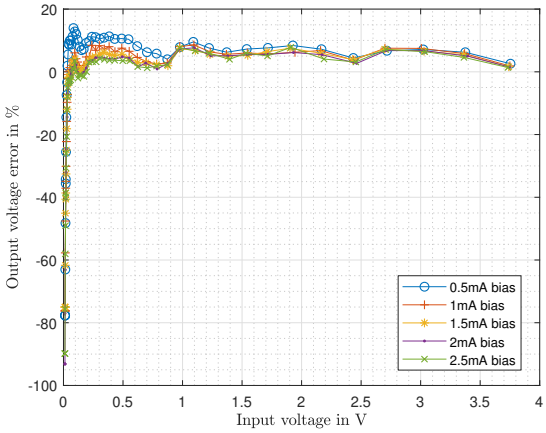


(d) 3ns Pulse-Width Small-Signals

Figure 4.13: PDH Effect of Tail Bias Current



(a) Measurement Error Small-Signals



(b) Measurement Error Full Range

Figure 4.14: PDH Feedback Current Analysis Measurement Error

Pulse-Width	1.5ns		2ns		3ns	
Tail-Current	10mA	15mA	10mA	15mA	10mA	15mA
Lower-Limit	116.3mV	110mV	73.5mV	73.2mV	48.1mV	47.5mV
Upper-Limit	1.327V	2.378V	2.406V	3.5V	3.5	3.5
Dynamic-Range	21.14dB	26.7dB	30.3dB	33.6dB	37.23dB	37.34dB

Table 4.1: PDH Using Bipolar Transistors Measurement Results Summary

In figure 4.15 the dynamic-range versus tail bias current is shown for the measured pulse-widths. The result shows, that the dynamic range for very short pulses can be increased significantly by increasing the bias current. Whereas for the 10ns long pulse the dynamic range is lowered with higher bias current, this is the case because the higher tail current leads to a higher voltage drop at the output of the OTA caused by the higher current through the clamping network.

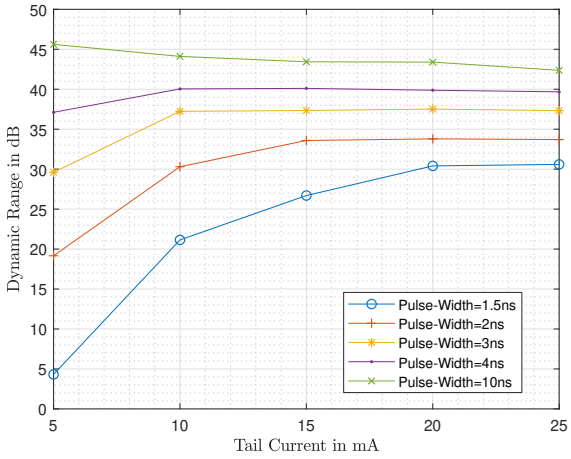


Figure 4.15: PDH Using Bipolar Transistors Dynamic-Range

The approximate current consumption of the circuit can be calculated by simply adding up the tail bias current, the feedback bias current and the supply current of the operational amplifier in voltage follower configuration. With the power consumption given by:

$$P \approx (I_{Tail} + I_{FB} + I_{opamp}) \cdot (V_{dd} - V_{ss}) \tag{4.5}$$

For the given circuit with $I_{FB} = 1.5mA$, $I_{opamp} = 0.5mA$, $V_{dd} = 5$ and $V_{ss} = -4$ resulting in:

$$P \approx I_{Tail} \cdot 9V + 18mW$$

Tail-Current	5mA	10mA	15mA	20mA	25mA
Power-Consumption	63mW	108mW	153mW	198mW	243mW

Table 4.2: PDH Using Bipolar Transistors Power Consumption

Performance Comparison

In table 4.3 existing circuits from literature are compared for the in this work relevant parameters. The comparison shows that the proposed circuit offers a much better minimum detectable pulse-width than existing circuits by a very low power consumption.

Topology	**Minimum detectable Pulse-Width**	**Dynamic-Range**	**Power Consumption**	**Design Goal**
Triggered Follow and Hold[9]	20ns	25dB	not specified (447mW Amplifiers)	Speed
Triggered Follow and Hold[10]	200ns	not specified	120mW	Low-Cost
Diode Capacitor configuration with Amplifier[13]	10ns	50dB	1.2W	Speed, Precision
Common-Base Configuration with Amplifier[14]	10ns	not specified	not specified	Speed, Precision
Common-Base Configuration with Amplifier[15]	not specified	46dB	not specified	Speed
Common-Base Configuration with Amplifier[16]	not specified	not specified	not specified	Precision
OTA with Current-Mirror[6]	500ns	not specified	200μW	Efficiency, CMOS Integration
OTA with Current-Mirror[17]	not specified	20dB	not specified	Precision
OTA with Current-Mirror[8]	200ns	20dB	6.6mW	Precision
Proposed Circuit	1.5ns 1.5ns 3ns	26.7dB 30dB 37.23B	108mW 198mW 108mW	Speed, Efficiency

Table 4.3: PDH Performance-Comparison

4.6 Conclusion

A new peak detect and hold circuit for nano-second pulses was designed and verified across different bias point settings. Measurement results for these different operating points were presented showing, that dynamic adjustment of the operating point can be used for matching the circuit performance to the requirements of a signal source, enabling maximum power efficiency. Pulses with a width of 3ns can be detected with a dynamic range of 37dB by a power consumption of only 108mW and a dynamic range of 26.7dB for pulse-widths of 1.5ns by a power consumption of 153mW.

Appendices

Appendix A

PDH using bipolar Transistors

A.1 Simulation Schematics

A.1.1 OTA Simulation

The simulations for the OTA were performed using a balanced- unbalanced-converter for applying the right common mode voltage and convert the input voltage to a differential voltage.

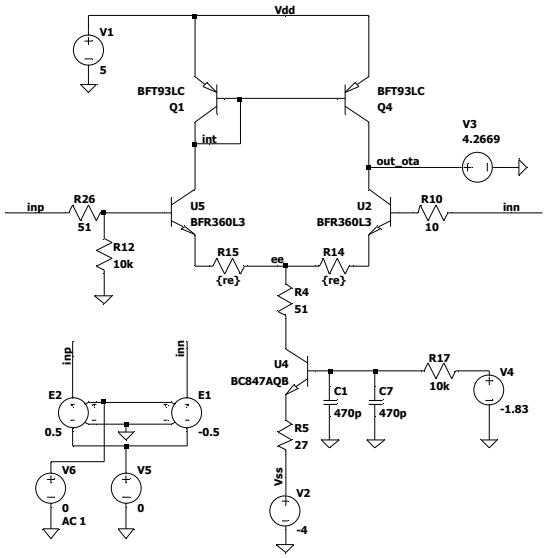


Figure A.1: PDH using bipolar Transistors OTA Simulation Schematic

A.1.2 Voltage Buffer Simulation

For the simulation of the voltage buffer a dc block capacitor in series with a very high resistor value was used to get only the ac content without offset. The load of the operational amplifier which is connected to the feedback is modeled by its input capacitance of 1pF.

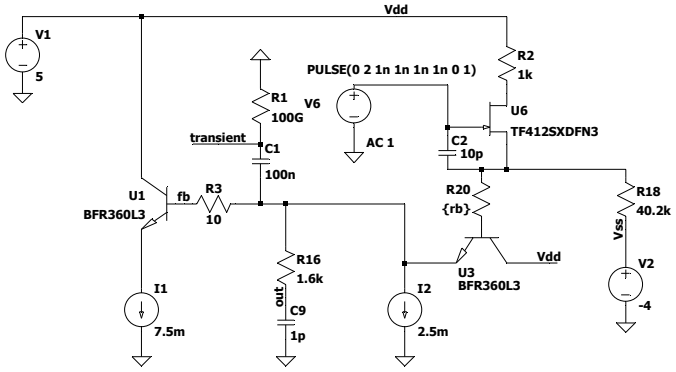


Figure A.2: PDH using bipolar Transistors Buffer Simulation Schematic

A.2 PCB Documents

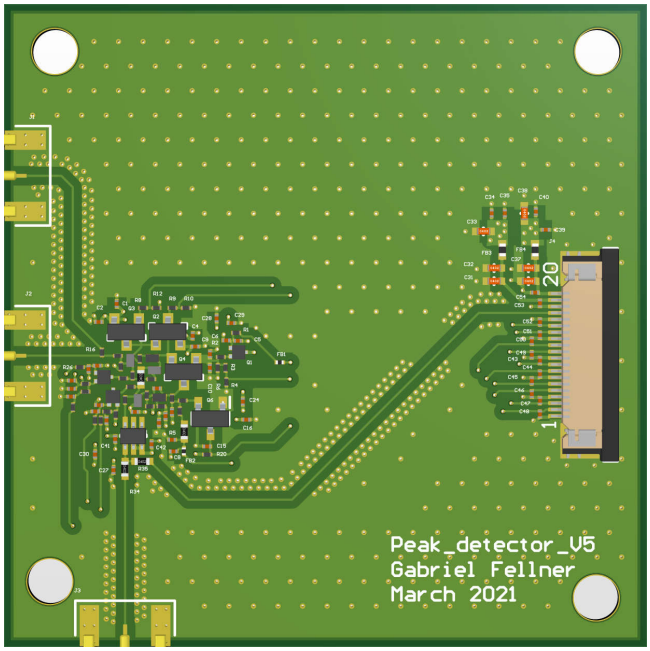


Figure A.3: PDH using bipolar Transistors PCB

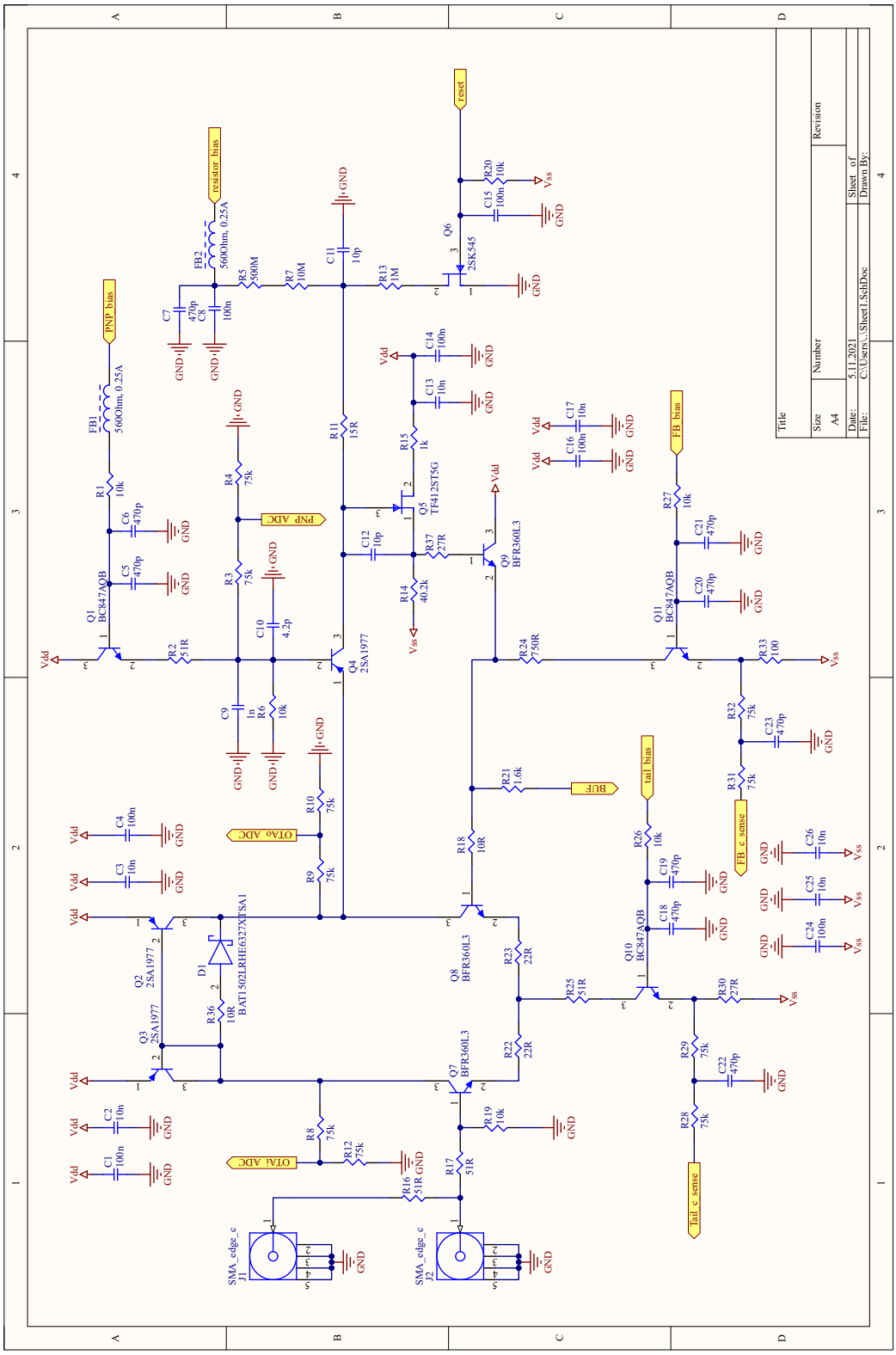


Figure A.4: PDH using bipolar Transistors Schematic Page 1

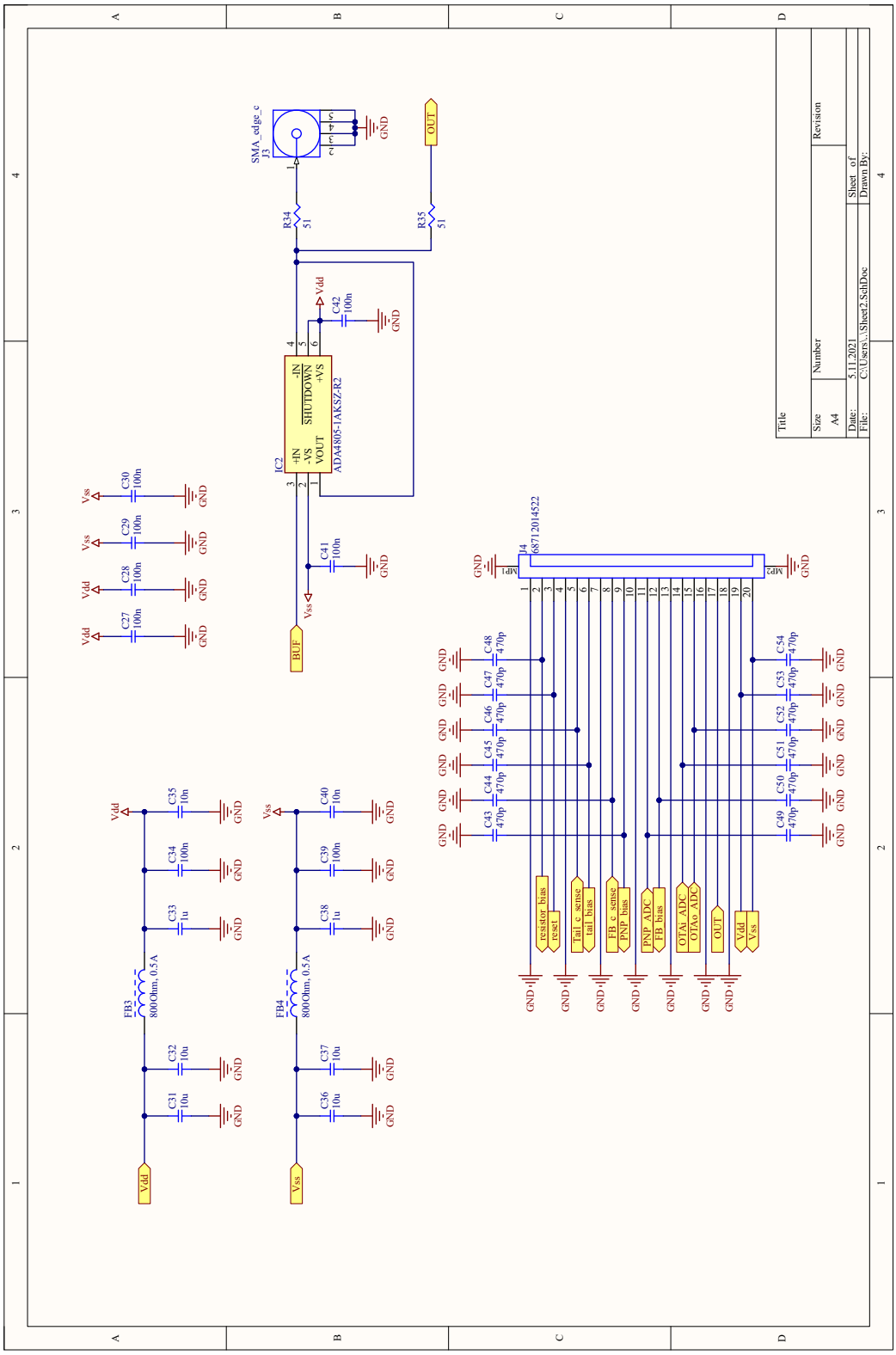


Figure A.5: PDH using bipolar Transistors Schematic Page 2

Appendix B

PDH using Operational Amplifier

In this chapter the design of the improved operational amplifier PDH circuit that was already discussed in the literature research is described. For testing the performance, a printed circuit board (PCB) was designed, manufactured and assembled.

B.1 Selection of Components and Design

B.1.1 Schematic

When the input voltage is greater than the output voltage U_1 starts to increase the voltage until the diode D_1 starts conducting and the hold capacitor is charged until the output voltage is equal to the input voltage. When the input-voltage decreases also the output voltage of U_1 decreases and therefore D1 stops conducting, but now D_2 starts conducting. This clamps the output voltage of U_1 one diode drop below the input voltage.

Because in the hold state the output voltage of U_1 is only one diode voltage lower than the input voltage (and not in the negative supply rail), the circuit is able to start into the detection phase very fast and the slew-rate requirement of the amplifier is mitigated [12] [11]. Bias indicates an option to apply a bias current to the diode. The component values and the exact schematic can be obtained from figure B.5.

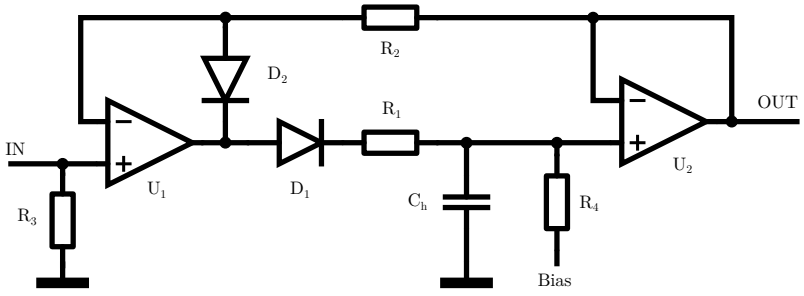


Figure B.1: PDH using Operational Amplifier Schematic

B.1.2 Operational amplifier

The most important requirements for this part are unity-gain stability, very high bandwidth, high slew-rate and low input bias current. OPA859 from Texas Instruments is one of the most promising parts for this application, the key-specifications are listed in the table below:

OPA859				
Parameter		Comment	Typ.	Unit
SSBW	Small-signal bandwidth	100mV_{pp}	1.8	GHz
LSBW	Large-signal bandwidth	$2V_{pp}$	400	MHz
SR	Slew-rate	2V step	1150	V/μs
t_r	Rise time	100mV step	0.3	ns
t_f	Fall time	100mV step	0.3	ns
I_{BN}, I_{BI}	Input-bias current	-	$\pm$0.5pA	MHz

Table B.1: OPA859 Specifications

Estimation of large signal rise time

With a first order approximation the fastest rise-time, where the circuit should be able to operate for large signals can be estimated by the equation:

$$t_{rLS} = \frac{0.35}{LSBW} \tag{B.1}$$

$$t_{rLS} = \frac{0.35}{400MHz} = 875ps$$

This result for the approximation fits the requirements very well.

Slew-rate limitation

The slew rate limits the rise time of the output voltage, the fastest possible rise-time can approximately calculated by the following equation:

$$t_{rSR} = \frac{0.8\Delta V}{SR} \tag{B.2}$$

The factor 0.8 is used because of the definition of the rise-time from 10% to 90%, ΔV is the voltage difference between the initial voltage at the output of the first amplifier and the maximum detectable peak voltage plus one diode drop. In worst case the output of the

first amplifier is one diode drop below the input voltage (for example if a small pulse comes right before a large pulse) and must increase to the peak input voltage plus the diode drop of the diode used for rectification. This means that ΔV is equal to two times the diode drop voltage plus the maximum peak voltage that should be detected. In the data sheet of the used operational amplifier the "Output voltage (high)" is only specified for a supply voltage of 3.3V with a minimum value of 2.3V. This would result in a maximum output voltage of 1.5V for supply with ±2.5V. Assuming a diode drop voltage of 250mV for the used schottky diodes we can estimate the fastest possible risetime at the hold capacitor. The maximum of the detectable peak voltage with this diode drop is therefor 1.25V.

$$t_{rSR} = \frac{0.8 \cdot (2 \cdot 0.25 + 1.25)}{1150V/\mu s} = 1.27ns$$

B.1.3 Calculation of hold capacitor

The maximum value of the hold capacitor is determined by the maximum output current of the operational amplifier and the highest possible slew-rate.

$$I_C = C\frac{\Delta U_C}{\Delta t} \tag{B.3}$$

$$C_{hold\,max} = \frac{I_C}{\frac{\Delta U_C}{\Delta t}}$$

$$C_{hold\,max} = \frac{I_{O\_LIN\,min}}{SR}$$

$$C_{hold\,max} = \frac{65mA}{1150V/\mu s} = 56.5pF$$

B.1.4 Diode selection

For this application the parallel capacitance and the forward voltage of the diode are the most important parameters. The forward voltage of the diode affects the difference in the output voltage that has to be generated by the amplifier and therefore challenges the slew-rate requirement of the op-amp. When the peak voltage at the hold capacitor is reached and the input voltage starts to decreases the diode will be reverse biased and the parallel capacitance of the diode is charged with the charge in the hold capacitor. This leads to a decrease in voltage at the hold capacitor, this voltage drop is usually known as pedestal voltage. By considering these two effects a schottky diode with very low parallel capacitance and forward voltage is chosen (BAT15-03W). This diode has a parallel capacitance of 0.28pF and a forward voltage of 350mV at a current of 10mA.

The voltage reduction because of the capacitance of the diode can be calculated by:

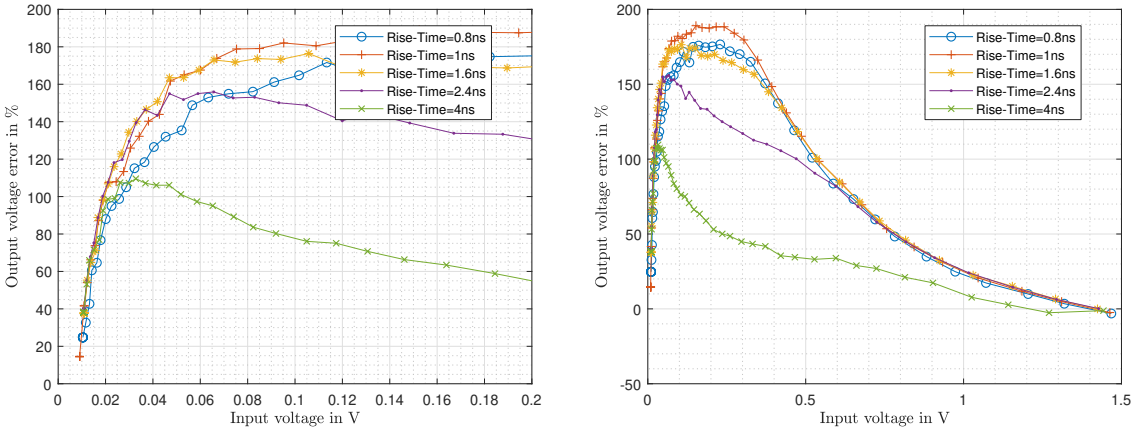
$$V_{error} = \Delta V \frac{C_{diode}}{C_{hold}} \tag{B.4}$$

For this circuit we can split this equation in one part which is always constant and one part which is a fixed percentage of the peak voltage:

$$V_{error} = (V_{diodedrop} + V_{peak}) \frac{C_{diode}}{C_{diode} + C_{hold}}$$
$$V_{error} = (0.35V + V_{peak}) \frac{0.28pF}{0.28pF + 47pF}$$
$$V_{error} = 0.0021 + 0.006 V_{peak}$$

B.2 Measurement Results

In figure B.2 the measurement error of the circuit is shown for the lower and full input voltage range by a variation of the rise-time. It shows that the circuit has a huge rise-time dependency and the output voltage is up to 2.9 times higher than the input voltage for a rise-time of 1ns. The strong effect of the rise-time is probably mainly caused by the large feedback delay in combination with quite high gain of the amplifier.

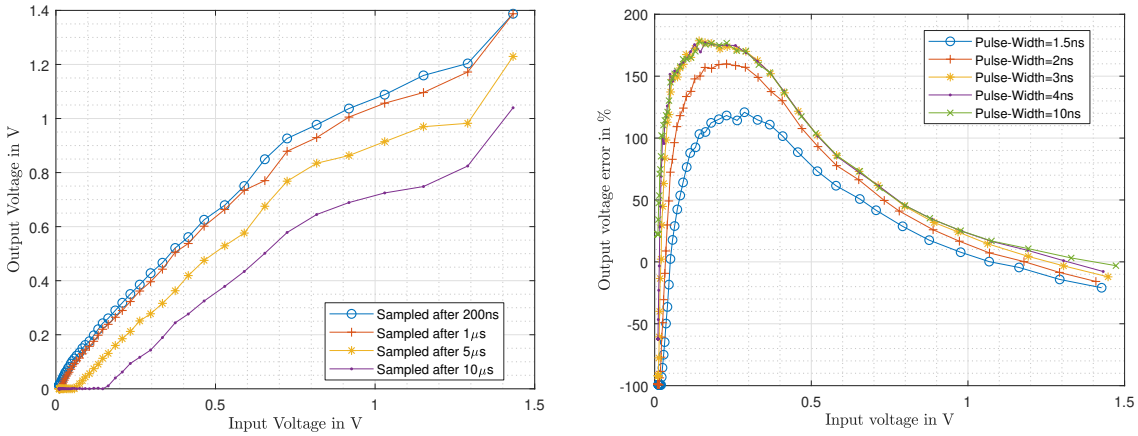


(a) Measurement Error Small-Signals

(b) Measurement Error full Range

Figure B.2: Opamp PDH Rise-Time Analysis Measurement Results

Figure B.3a shows of course also the high error because of the rise-time of 4ns used in the measurement. From the plot also a very high droop-rate can be obtained, this problem could easily be fixed by a reduction of bias current, but was not relevant at this time. From B.3b one can see, that the error for pulse-widths larger as 2ns is very similar.



(a) Long-Time Analysis

(b) Pulse-Width Analysis Error

Figure B.3: Opamp PDH Measurement Results

B.3 Conclusion

The measurement results revealed, that the circuit is very fast but suffers very large not acceptable rise-time dependency. By connecting the input of the buffer for the feedback to the cathode of the diode for rectification. This would result in a low-pass for the voltage across the hold capacitor and increase the amount of feedback voltage for high current. Another way would be including a rise-time filter before the hold capacitor instead of a simple low-pass filter, this would allow to determine a minimum rise-time for the voltage of the hold capacitor.
Anyway this structure of circuit was not longer analyzed because of the high power consumption of 20.5mA per amplifier at 5V.

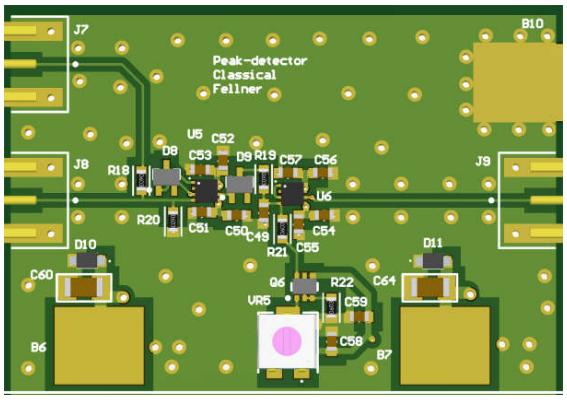


Figure B.4: PDH using operational Amplifier PCB

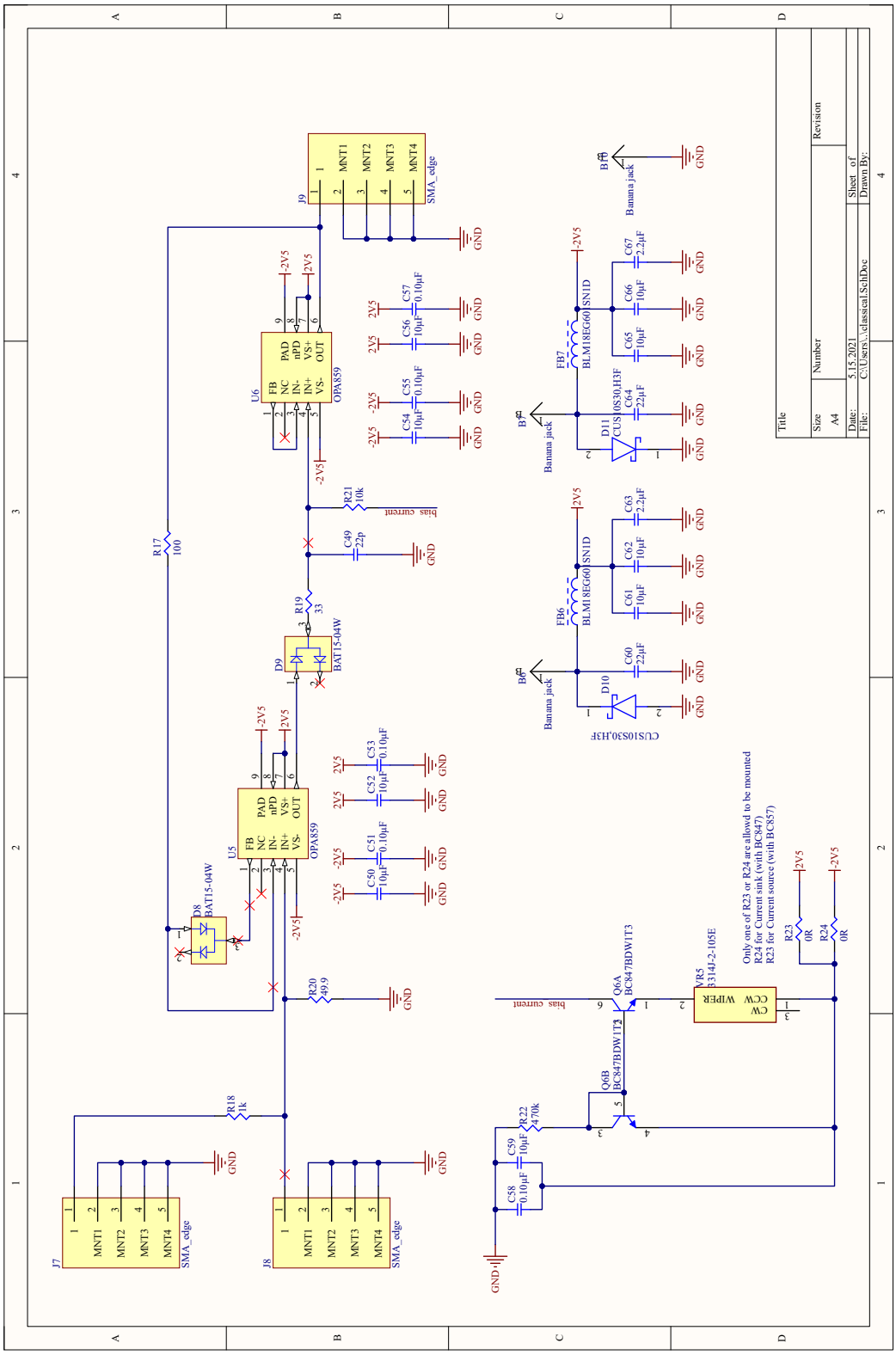


Figure B.5: PDH using Operational Amplifier Schematic Page 1

Appendix C

Integrated CMOS PDH

C.0.1 Schematic

For a first try and performance estimation a PDH circuit was simulated in Cadence Virtuoso using a 180nm technology. For the design of the circuit g_m over I_d technique was used. Therefor the whole process was characterized in simulation.

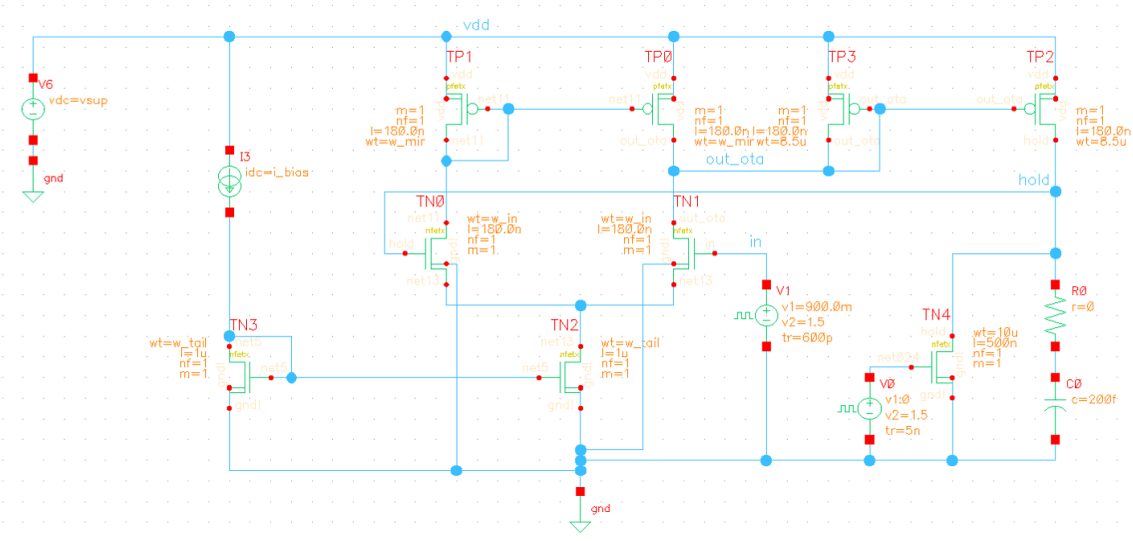


Figure C.1: CMOS PDH Schematic

C.0.2 Design of the Circuit

For the design of the circuit a load capacitance of 100fF was assumed and the OTA was designed for a gain bandwidth product (GBWP) of 5GHz. At the beginning the required transconductance of the input differential pair MOSFET was calculated.

$$GBWP = \frac{g_{min}}{2\pi C_L} \tag{C.1}$$

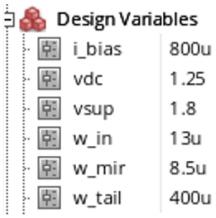


Figure C.2: CMOS PDH Variable Values

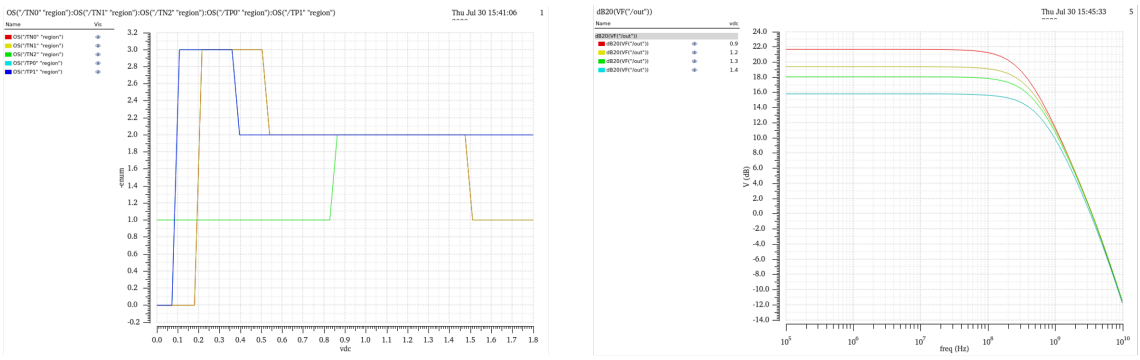
Which results in 3.15mS. For g_m/I_d a value of 8 is used for getting small dimensions of the device to reduce parasitic capacitance.

$$I_d = \frac{g_{min}}{\frac{g_m}{I_d}} \tag{C.2}$$

Using this g_m/I_d the required bias current for one transistor of the input differential pair results to 400μA. By using the data generated in the characterization of the technology (I_d/W vs g_m/I_d) g_m/W is determined to 31, resulting in a width of 13um.

C.0.3 Simulation Results

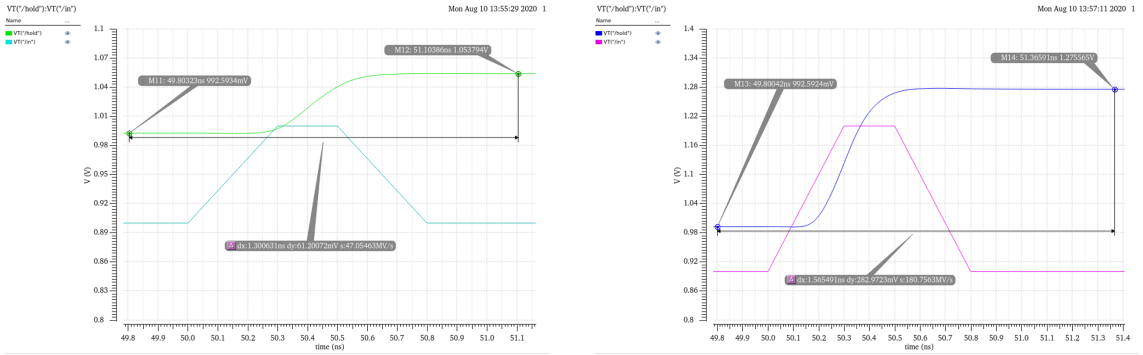
Figure C.3a shows that the input common-mode range is very low ranging from 900mV to 1.4V. This is because of the very low g_m over I_d of 8. The ac analysis in figure C.3b shows that the intended GBWP of 5GHz was not reached, the simulated GBWP is only 3.2GHz, probably because of tho high parasitic capacitance.



(a) Input Common-Mode Range (b) AC Characteristic

Figure C.3: CMOS PDH Simulation Result

Figure C.4 and figure C.5 show the full circuit response to a 500ps wide pulse with a rise- and fall-time of 300ps. To get the right common mode voltage an offset was added to the signal. The result shows that a voltage difference of only 61mV was reached for the 100mV, 283mV for the 300mV pulse and 602mV for the 600mV pulse.



(a) 100mV Amplitude (b) AC Characteristic

Figure C.4: CMOS PDH Transient Simulation Result

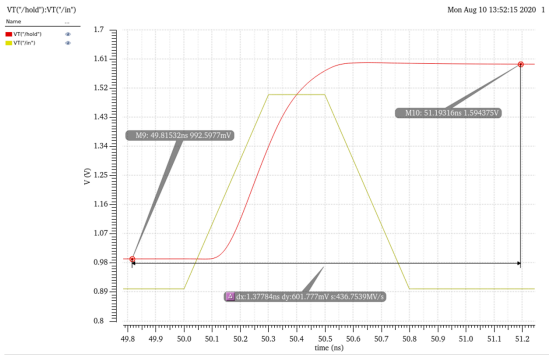


Figure C.5: CMOS PDH Transient Simulation Result 600mV

A major problem which can be obtained is the high offset voltage already before the pulse. This is a result of the limited gain of the OTA.

C.0.4 Conclusion

First simulation shows that the circuit concept can be very fast, but there are still problems with the limited gain which causes pulse-width dependent offset to the result. For sure this issues can be minimized by experienced analog design engineers.
The concept of this CMOS PDH was no longer scope of this thesis because of the high complexity and lead time of designing an IC.

Appendix D

ATE Python Program

D.1 Global Parameters and Communication Setup

The program starts with importing all required libraries which are used.

```
import numpy as np
import math
from scipy import signal
import matplotlib.cm as cm
import matplotlib.mlab as mlab
import matplotlib.pyplot as plt
import time
import pyvisa as visa
import struct
from RsInstrument import *
import serial
import serial.tools.list_ports
import nidaqmx
from nidaqmx.constants import TerminalConfiguration
```

In the next step global communication interface parameters are defined for the programmable attenuators and oscilloscope.

```
global rm
global dmm_AWG
#Serial interface global variables for programmable attenuators
global ser_a
global ser_b
#Communication parameters for oscilloscope
GPIB_AWG = 'TCPIP0::192.168.0.23::inst0::INSTR'
rm = visa.ResourceManager('@py')
```

For establishing a connection to the oscilloscope and for setting initial parameters the function RTO_initialization() is used.

```
    def RTO_initialization():
        global rto
        rto = None
        try:
            rto = RsInstrument('TCPIP::192.168.0.101::INSTR', True,
            ↪ True)
            rto.assert_minimum_version('1.8.0')
            rto.visa_timeout = 5000  # Timeout for VISA Read Operations
            rto.opc_timeout = 15000  # Timeout for opc-synchronised
            ↪ operations
            rto.instrument_status_checking = True  # Error check after
            ↪ each command
        except Exception as ex:
            print('Error initializing the instrument session:\n' +
            ↪ ex.args[0])
            exit()

        print(f'rto2000 IDN: {rto.idn_string}')
        print(f'rto2000 Options: {",".join(rto.instrument_options)}')

        rto.clear_status()
        rto.reset()
        rto.write_str("SYST:DISP:UPD ON")  # Display update ON - switch
        ↪ OFF after debugging

        # Basic Settings:
        rto.write_str("ACQ:POIN:AUTO RESolution")  # Define Horizontal
        ↪ scale by number of points
        rto.write_str("ACQ:RES 10e-12")  # xs Resolution

        rto.write_str("TIMebase:REFerence 10") #sets horizontal offset
        ↪ to x% of screen
        #rto.write_str("ACQuire:POINts 1e+6") # TODO
        rto.write_str("TIM:RANG 0.00001")  # 10ms Acquisition time

        #Channel 1
        rto.write_str("CHAN1:COUP DC")  # Coupling AC, LIMIT (DC 1MOhm)
        ↪ #DC (50ohm)
        rto.write_str("CHAN1:STAT ON")  # Switch Channel 1 ON
        rto.query_opc()

        #Channel 2
        rto.write_str("CHAN2:COUP DC")  # Coupling AC, LIMIT (DC 1MOhm)
        ↪ #DC (50ohm)
        rto.write_str("CHAN2:STAT ON")  # Switch Channel 2 ON
        rto.query_opc()
```

```
38
39          # Trigger Settings:
40          rto.write_str("TRIG1:MODE NORMal")  # Trigger Auto mode in case
            ↪ of no signal is applied
41          rto.write_str("TRIG1:SOUR CHAN2")  # Trigger source CH1
42          rto.write_str("TRIG1:TYPE EDGE;:TRIG1:EDGE:SLOP POS")  # Trigger
            ↪ type Edge Positive
43          rto.write_str("TRIG1:LEV1 0.02")  # Trigger level 20mV
44          rto.write_str("RUN")
45          rto.query_opc()  # Using *OPC? query waits until all the
            ↪ instrument settings are finished
46
47          pass
```

With ATT_init() the connection to the programmable attenuators is established.

```
1       def ATT_init():
2           global ser_a
3           global ser_b
4
5           # open serial port
6           ser_a = serial.Serial()
7           ser_b = serial.Serial()
8           #port_a, port_b = ATT_find()
9           ser_a.port = "COM4"
10          ser_b.port = "COM5"
11          # ser.port = "/dev/ttyUSB0"
12          ser_a.baudrate = 9600
13          ser_a.bytesize = serial.EIGHTBITS  # number of bits per bytes
14          ser_a.parity = serial.PARITY_NONE  # set parity check: no parity
15          ser_a.stopbits = serial.STOPBITS_ONE  # number of stop bits
16          ser_a.dsrdtr = False
17          ser_a.rtscts = False
18          ser_a.xonxoff = False
19          ser_a.dtr = False
20          ser_a.rts = False
21
22          ser_b.baudrate = 9600
23          ser_b.bytesize = serial.EIGHTBITS  # number of bits per bytes
24          ser_b.parity = serial.PARITY_NONE  # set parity check: no parity
25          ser_b.stopbits = serial.STOPBITS_ONE  # number of stop bits
26          ser_b.dsrdtr = False
27          ser_b.rtscts = False
28          ser_b.xonxoff = False
29          ser_b.dtr = False
30          ser_b.rts = False
31
```

```
32          ser_a.open()
33          time.sleep(1.5)  # needed as arduino is reseted on connect
34          ser_a.flush()
35          ser_a.flushInput()
36          ser_a.flushOutput()
37
38          ser_b.open()
39          time.sleep(1.5)  # needed as arduino is reseted on connect
40          ser_b.flush()
41          ser_b.flushInput()
42          ser_b.flushOutput()
43
44          pass
```

The initialization of the arbitrary waveform generator with the desired waveform and sampling frequency is done by AWG_init(wfmData, fs). Where the functions for creating the marker data and waveform message are shown below.

```
1       def AWG_init(wfmData, fs):
2           global rm
3           global dmm_AWG
4           dmm_AWG = rm.open_resource(GPIB_AWG)
5           dmm_AWG.encoding = 'latin_1'
6           dmm_AWG.timeout = 25000
7           dmm_AWG.write_termination = None
8           dmm_AWG.read_termination = '\n'
9
10          #Reset AWG
11          dmm_AWG.write('*RST')
12          dmm_AWG.query('*opc?')
13          dmm_AWG.write('*CLS')
14          dmm_AWG.query('*opc?')
15
16          #Name of waveform file on AWG
17          name = 'Waveform_from_control_PC'
18
19          #Set marker data
20          markerData = createrandomMarkerData(wfmData)
21
22          #Create waveform on AWG with waveform data and marker data
23          msg = makewaveformDataMessage(name, wfmData, markerData)
24          print(msg)
25          dmm_AWG.write_raw(msg)
26          dmm_AWG.query('*opc?')
27
28
29          # Load waveform, being playback, and turn on output
```

```
30          dmm_AWG.write(f'SOURce2:WAVeform "{name}"')
31          #dmm_AWG.write('AWGCONTROL:RUN:IMMediate')
32          dmm_AWG.write(f'SOUR1:FREQ {fs}')
33          #dmm_AWG.write('SOUR2:VOLT 1')
34          dmm_AWG.write('AWGControl:RUN')
35          dmm_AWG.write('AWGCONTROL:RMODE TRIGGERED')
36          #dmm_AWG.write('AWGC:RMOD CONT')
37          dmm_AWG.query('*opc?')
38          dmm_AWG.write('OUTPut2 ON')
39          dmm_AWG.query('*opc?')
40
41          pass
```

Because marker data was not required for this operation the function createrandomMArkerData(wfmData) was used to generate random markers.

```
1  def createrandomMarkerData(wfmData):
2      # Create random Marker Data
3      # Marker data is an 8 bit value. Bit 6 is marker 1 and bit 7 is
       ↪ marker 2
4      recordLength = len(wfmData)
5      exData1 = (1 << 6) * np.random.randint(2, size=recordLength,
       ↪ dtype=np.uint8)
6      exData2 = (1 << 7) * np.random.randint(2, size=recordLength,
       ↪ dtype=np.uint8)
7      markerData = exData1 + exData2
8
9      return markerData
```

In makewaveformDataMessage(...) the required binblock header and binbloc data is generated for sending to the AWG.

```
1  def makewaveformDataMessage(name, wfmData, markerData):
2      # Create Data for binblock
3      recordLength = len(wfmData)
4      datawmarker = np.row_stack((wfmData, markerData))
5      binblockData = makeWFMXFileBinaryData(datawmarker, 1)
6      binblockHeader = makebinBlockHeader(binblockData)
7
8      #Create waveform on AWG with waveform data
9      dmm_AWG.write('WLISt:WAVeform:NEW "{}", {}, REAL'.format(name,
       ↪ recordLength))
10     dmm_AWG.query('*opc?')
11     msg = f'WLISt:WAVeform:DATA "{name}", 0,
       ↪ {recordLength}'.encode('ascii') + binblockHeader + binblockData
12
13     return msg
```

```
def makebinBlockHeader(binblockData):
    len_file = len(binblockData)
    len_str = len(str(len_file))  # No. of digits needed to write length
    size_str = (f', #{len_str}{len_file}').encode('ascii')

    return size_str
```

```
def makeWFMXFileBinaryData(data: np.ndarray, amplitude: float) -> bytes:
    """
    For the binary part.
    Note that currently only zero markers or two markers are supported;
    one-marker data will break.
    Args:
        data: Either a shape (N,) array with only a waveform or
            a shape (M, N) array with waveform, marker1, marker2,
↪   marker3, i.e.
            data = np.array([wfm, m1, ...]). The waveform data is
↪   assumed
            to be in V.
        amplitude: The peak-to-peak amplitude (V) assumed to be set on
↪   the
            channel that will play this waveform. This information is
            needed as the waveform must be rescaled to (-1, 1) where
            -1 will correspond to the channel's min. voltage and 1 to
↪   the
            channel's max. voltage.
    """

    channel_max = amplitude
    channel_min = -amplitude

    shape = np.shape(data)

    if len(shape) == 1:
        N = shape[0]
        binary_marker = b''
        wfm = data
    else:
        N = shape[1]
        M = shape[0]
        wfm = data[0, :]
        markers = data[1, :]
        for i in range(1, M-1):
            markers += data[i+1, :] * (2**i)
        markers = markers.astype(int)
        fmt = N*'B'  # endian-ness doesn't matter for one byte
```

```
36
37          binary_marker = struct.pack(fmt, *markers)
38
39      if wfm.max() > channel_max or wfm.min() < channel_min:
40          print('Waveform exceeds specified channel range.'
41                      ' The resulting waveform will be clipped. '
42                      'Waveform min.: {} (V), waveform max.: {} (V),'
43                      'Channel min.: {} (V), channel max.: {} (V)'
44                      ''.format(wfm.min(), wfm.max(), channel_min,
                        ↪  channel_max))
45
46      # the data must be such that channel_max becomes 1 and
47      # channel_min becomes -1
48      scale = 2/amplitude
49      wfm = wfm*scale
50
51      # TODO: Is this a fast method?
52      #fmt = '<' + N*'f'
53      #binary_wfm = struct.pack(fmt, *wfm)
54      #binary_out = binary_wfm + binary_marker
55
56      fmt = '<' + 'f'
57      binary_wfm = struct.pack(fmt, wfm[0]) + struct.pack('B', markers[0])
58      for number in range(1,len(wfm)):
59          binary_wfm = binary_wfm + struct.pack(fmt, wfm[number]) +
            ↪  struct.pack('B', markers[number])
60
61
62      #binary_wfm = struct.pack(fmt, *wfm)
63      #binary_out = binary_wfm + binary_marker
64      binary_out = binary_wfm
65      return binary_out
```

D.2 Setup Configuration

For setting the pulse amplitude mainly the programmable attenuators are used. The attenuation of the attenuators is set by ATT_a_set(att) and ATT_b_set(att).

```
1       def ATT_a_set(att):
2           #set att
3           dB=att
4           send_cmd="#"+str(dB)+"\n"
5           ser_a.write(bytes(send_cmd, 'utf-8'))
6           pass
```

```
1       def ATT_b_set(att):
```

```
        #set att
        dB=att
        send_cmd="#"+str(dB)+"\n"
        ser_b.write(bytes(send_cmd, 'utf-8'))
        pass
```

The real intended attenuation is set with ATT_set(att) which adjusts both programmable attenuators and the amplitude set by de AWG.

```
    def ATT_set(att):
        if att <= 15:
            ATT_a_set(att)
            ATT_b_set(0)
            dmm_AWG.write('SOUR2:VOLT 1')
            dmm_AWG.query('*opc?')
        elif att > 15 and att <= 47:
            ATT_a_set(15)
            ATT_b_set(att-15)
            dmm_AWG.write('SOUR2:VOLT 1')
            dmm_AWG.query('*opc?')
        elif att > 47:
            ATT_a_set(15)
            ATT_b_set(32)
            awg_amp = 1 * 10**(-(att-47)/20)
            dmm_AWG.write(f'SOUR2:VOLT {awg_amp}')
            dmm_AWG.query('*opc?')

        time.sleep(0.1)
        pass
```

RTO_set_vertical_scale(...) sets the trigger and the channel scaling of the scope to suitable values for the expected amplitudes.

```
    def RTO_set_vertical_scale(amplitude, CH2_probe_factor):
        #Sets vertical range of scope to 40% more than pulse amplitude
        #Offset set to have 10% space to lower limit and 30% to upper
        ↪ limit
        volts_per_div = np.array([0.005, 0.01, 0.02, 0.04, 0.05, 0.1,
        ↪ 0.2, 0.4, 0.5, 1])
        avail_range = volts_per_div * 10    #With 10 divisions vertical

        #trigger_ch1 = 0.25 * amplitude + 0.001
        #range_ch1 = min(avail_range[avail_range > (amplitude *
        ↪ 4/2.02)])

        if amplitude/2 <= 0.4:
            range_ch1 = min(avail_range[avail_range > (amplitude * 2 /
            ↪ 2.02+0.025)])
```

```
            offset_ch1 = range_ch1 / 2 - 0.025
        else:
            range_ch1 = min(avail_range[avail_range > (amplitude * 2 /
            ↪  2.02 + 0.1)])
            offset_ch1 = range_ch1 / 2 - 0.1

        #offset_ch1 = range_ch1 / 2 - range_ch1 * lower_space

        #CH1
        rto.write_str(f"CHAN1:RANG {range_ch1}")  # Horizontal range
        rto.query_opc()
        rto.write_str(f"CHANnel1:OFFSet {offset_ch1}")  # Offset
        rto.query_opc()
        #rto.write_str(f"TRIG1:LEV1 {trigger_ch1}")
        #rto.query_opc()

        #CH2 with probe resistor
        trigger_ch2 = 0.3 * amplitude * CH2_probe_factor + 0.004
        range_ch2 = range_ch1 #min(avail_range[avail_range > amplitude *
        ↪  4 * CH2_probe_factor])
        offset_ch2 = offset_ch1
        #offset_ch2 = range_ch2 / 2 - range_ch2 * lower_space

        rto.write_str(f"CHAN2:RANG {range_ch2}")
        rto.query_opc()
        rto.write_str(f"CHANnel2:OFFSet {offset_ch2}")
        rto.query_opc()
        rto.write_str(f"TRIG1:LEV2 {trigger_ch2}")
        rto.query_opc()
```

D.3 Functions for Circuit adjustment

In ni_ai_init(task) the configuration of the used analog inputs of the ni-IO-card is set to single ended configuration.

```
    def ni_ai_init(task):
        task.ai_channels.add_ai_voltage_chan("Dev1/ai0",
        ↪  terminal_config=TerminalConfiguration.RSE)
        task.ai_channels.add_ai_voltage_chan("Dev1/ai1",
        ↪  terminal_config=TerminalConfiguration.RSE)
        task.ai_channels.add_ai_voltage_chan("Dev1/ai2",
        ↪  terminal_config=TerminalConfiguration.RSE)
        task.ai_channels.add_ai_voltage_chan("Dev1/ai3",
        ↪  terminal_config=TerminalConfiguration.RSE)
        task.ai_channels.add_ai_voltage_chan("Dev1/ai4",
        ↪  terminal_config=TerminalConfiguration.RSE)
```

```
7           pass
```

The analog outputs are initialized by ni_ao_init(task, device), where device is required because the DACs of two ni-IO-cards are used.

```
1       def ni_ao_init(task, device):
2           task.ao_channels.add_ao_voltage_chan(f'{device}/ao0')
3           task.ao_channels.add_ao_voltage_chan(f'{device}/ao1')
4           pass
```

Reading of analog values is done using ni_read_data(samples), where the output is an array of analog values with size of samples.

```
1       def ni_read_data(samples):
2           with nidaqmx.Task() as task:
3               ni_ai_init(task)
4               data = task.read(number_of_samples_per_channel=samples)
5               return data
```

Writing data to the analog outputs is done with ni_write_data(device, ao_0, ao_1).

```
1       def ni_write_data(device, ao_0, ao_1):
2           with nidaqmx.Task() as task:
3               ni_ao_init(task, device)
4               task.write([[ao_0],[ao_1]])
5               pass
```

The function set_fb_current(...) sets the feedback current of the circuit to the desired value by reading the voltage across the current sensing resistor and in relation to that increasing or decreasing the bias voltage of the transistor used as a current sink.

```
1       def set_fb_current(fb_bias_v, desired_fb_current, pnp_init_v,
        ↪  samples):
2           dac_lsb = 20 / (2 ** 14)  # for LSB steps
3           ni_write_data('Dev2', pnp_init_v, fb_bias_v)
4
5
6           data = np.array(ni_read_data(samples))
7           fb_current_meas = (np.mean(data[1,:])+4)/100
8
9           iterations = 0
10          while (fb_current_meas < (desired_fb_current - 0.0001)):
11              iterations +=1
12              if iterations > 16000 or fb_bias_v > 5:
13                  print(f'Error during FB current adjustment, FB bias
                    ↪  {fb_bias_v}V')
14                  break
15              fb_bias_v = fb_bias_v + dac_lsb  # LSB steps
16              ni_write_data('Dev2', pnp_init_v, fb_bias_v)
17              data = np.array(ni_read_data(samples))
```

```
            fb_current_meas = (np.mean(data[1,:])+4)/100

        iterations = 0
        while (fb_current_meas > (desired_fb_current + 0.0001)):
            iterations +=1
            if iterations > 16000 or fb_bias_v > -4:
                print(f'Error during FB current adjustment, FB bias
                ↪ {fb_bias_v}V')
                break
            fb_bias_v = fb_bias_v - dac_lsb  # LSB steps
            ni_write_data('Dev2', pnp_init_v, fb_bias_v)
            data = np.array(ni_read_data(samples))
            fb_current_meas = (np.mean(data[1,:])+4)/100

        print(f'FB bias current set to: {fb_current_meas}A')
        return fb_bias_v
```

For adjusting the circuit to the desired tail bias current the function set_tail_current(...) is used. The principle of the operation is very similar to the function for setting the feedback bias current.

```
    def set_tail_current(tail_curr_bias_v, desired_tail_current,
    ↪ bias_res_v, samples):
        dac_lsb = 20/(2**14)  # for LSB steps

        ni_write_data('Dev1', bias_res_v, tail_curr_bias_v)

        data = np.array(ni_read_data(samples))
        tail_current_meas = (np.mean(data[0, :]) + 4) / 27


        iterations = 0
        while (tail_current_meas < (desired_tail_current - 0.0001)):
            iterations +=1
            if iterations > 16000 or tail_curr_bias_v > 5:
                print(f'Error during tail current adjustment, tail bias
                ↪ {tail_curr_bias_v}V')
                break
            tail_curr_bias_v = tail_curr_bias_v + dac_lsb  # LSB steps
            ni_write_data('Dev1', bias_res_v, tail_curr_bias_v)
            data = np.array(ni_read_data(samples))
            tail_current_meas = (np.mean(data[0, :]) + 4) / 27


        while (tail_current_meas > (desired_tail_current + 0.0001)):
            iterations += 1
            if iterations > 16000 or tail_curr_bias_v < -4:
```

```
25                  print(f'Error during tail current adjustment, tail bias
                    ↪  {tail_curr_bias_v}V')
26                  break
27              tail_curr_bias_v = tail_curr_bias_v - dac_lsb  # LSB steps
28              ni_write_data('Dev1', bias_res_v, tail_curr_bias_v)
29              data = np.array(ni_read_data(samples))
30              tail_current_meas = (np.mean(data[0, :]) + 4) / 27
31
32          print(f'Tail bias current set to: {tail_current_meas}A')
33          return tail_curr_bias_v
```

set_pnp_bias_voltage(...) sets the bias voltage of the pnp-transistor in common-base configuration to the right value to achieve a revers bias of the diode in the clamping network of 50mV.

```
1       def set_pnp_bias_voltage(pnp_bias_v, fb_bias_v, samples):
2           dac_lsb = 20 / (2 ** 14)  # for LSB steps
3
4           data = np.array(ni_read_data(samples))
5           DC_ota_out = np.mean(data[4, :])*2
6           DC_ota_intern = np.mean(data[3, :])*2
7           DC_ota_diff_out = DC_ota_out - DC_ota_intern
8
9           iterations = 0
10          while(DC_ota_diff_out < 0.200):
11              iterations += 1
12              if iterations > 16000 or pnp_bias_v > 5:
13                  print(f'Error during pnp bias adjustment, poti output
                    ↪  {pnp_bias_v}V')
14                  break
15              pnp_bias_v = pnp_bias_v + dac_lsb  # lsb steps
16              ni_write_data('Dev2', pnp_bias_v, fb_bias_v)
17              data = np.array(ni_read_data(samples))
18              DC_ota_out = np.mean(data[4, :])*2
19              DC_ota_intern = np.mean(data[3, :])*2
20              DC_ota_diff_out = DC_ota_out - DC_ota_intern
21
22          while(DC_ota_diff_out >= 0.050):
23              iterations += 1
24              if iterations > 16000 or pnp_bias_v < 0:
25                  print(f'Error during pnp bias adjustment, poti output
                    ↪  {pnp_bias_v}')
26                  break
27              pnp_bias_v = pnp_bias_v - dac_lsb  # lsb steps
28              ni_write_data('Dev2', pnp_bias_v, fb_bias_v)
29              data = np.array(ni_read_data(samples))
30              DC_ota_out = np.mean(data[4, :])*2
```

```
            DC_ota_intern = np.mean(data[3, :])*2
            DC_ota_diff_out = DC_ota_out - DC_ota_intern

        print(f'OTA output difference set to: {DC_ota_diff_out}V')
        return pnp_bias_v
```

In set_circuit_op(tail_current_mA) all functions for adjusting operating point parameters of the circuit are combined to set the circuit in the right operating point for the desired tail current.

```
    def set_circuit_op(tail_current_mA):
        #Set circuit to op
        #Init values
        fb_bias_v_init = -3.34
        pnp_bias_v_init = 4.6
        tail_curr_bias_v_init = -3.1

        samples = 1
        fb_current = 0.0025
        tail_current = tail_current_mA/1000
        bias_res_v = -1

        init_circuit(fb_bias_v_init, pnp_bias_v_init,
        ↪ tail_curr_bias_v_init, bias_res_v)
        fb_bias_v = set_fb_current(fb_bias_v_init, fb_current,
        ↪ pnp_bias_v_init, samples)
        tail_curr_bias_v = set_tail_current(tail_curr_bias_v_init,
        ↪ tail_current, bias_res_v, samples)
        pnp_bias_v = set_pnp_bias_voltage(pnp_bias_v_init, fb_bias_v,
        ↪ samples)

        pass
```

D.4 Waveform Generation

For generating the sampled waveform for the AWG the function trap_pulse(...) is used which generates a trapezoidal pulse. Depending on the used amplifier pulse-predistorion can be included.

```
def trap_pulse(amplitude, tr, duration, tf, fs_awg):
    #Times must be given in ns

    tr = int(np.rint(tr/0.8 * fs_awg / 1e9))    #conversion for ns, div
    ↪ by 0.8 because risetime is from 10% to 90%
    tf = int(np.rint(tf/0.8 * fs_awg / 1e9))
    ontime = int(np.rint(duration * fs_awg / 1e9 - (tr+tf)/2))
    periode = tr + ontime + tf
```

```
8
9       t = np.linspace(0, periode, periode+1)
10      pulse = np.zeros(periode+1)
11      # Create one period of the trapezoidal pulse waveform
12      for number in range (0, periode+1):
13          if t[number] <= tr:
14              pulse[number] = amplitude * t[number]/tr
15          elif (t[number] > tr and t[number] <= ontime + tr):
16              pulse[number] = amplitude
17          elif (t[number] > tr + ontime and t[number] < tr + ontime + tf):
18              pulse[number] = amplitude - amplitude /tf * (t[number] - (tr
                ↪  + ontime))
19          else:
20              pulse[number] = 0
21
22      # Add zero padding before/after pulse, important for filter
23      pulse = np.pad(pulse, (10, 10), 'constant', constant_values=(0, 0))
24
25      #pulse = pulse_predistortion_inv_hp(pulse, fs_awg)
26
27      return pulse
```

For the BONN amplifier the predistortion-function pulse_predistortion_inv_hp(...) is used.

```
1       def pulse_predistortion_inv_hp(pulse, fs):
2           b, a = signal.butter(1, 8e6, 'high', fs=fs) # Highpass filter
            ↪  coefficients
3           dist_pulse = signal.lfilter(a, b, pulse)    # Apply inverse
            ↪  highpas filter
4
5           #w, h = signal.freqz(a, b)
6           #plt.plot(w*fs/(2*np.pi), 20 * np.log10(abs(h)))
7           #plt.grid(True)
8           #plt.show()
9
10          return dist_pulse
```

For triggering a pulse of the AWG the function AWG_trig() is used.

```
1       def AWG_trig():
2           time.sleep(1)   # Necessery to wait at least 1s to get trigger
3           dmm_AWG.write('*TRG')
4           pass
```

D.5 Data Processing

After a pulse of the AWG was triggered, the peak values are measured by this function. The function therefor filters the signal and measures offset voltages for getting the peak value of the input voltage an measuring the output voltage after a certain amount of time.

```
1       def RTO_read_CH1_max_and_CH2_after_ns(measure_delay,
        ↪  CH2_probe_factor):
2           measure_delay = measure_delay / 1e9     #rounding problem with
            ↪  negative potentials e.g. 1e-9
3
4           rto.write_str("STOP;")
5           rto.query_opc()
6           header_str_ch2 = rto.query_str('CHANnel1:DATA:HEADer?')     #
            ↪  Header sometimes makes problems restart scope
7           rto.query_opc()
8
9           header = np.fromstring(header_str_ch2, dtype=float, sep=',')
10
11          #rto.write_str("EXPort:WAVeform:INCXvalues OFF")     #
            ↪  interleaved xy on/off
12          data_ch1 = rto.query_str('CHANnel1:DATA:VAL?')  # some non ascii
            ↪  bytes causing troubles
13          rto.query_opc()
14          data_ch2 = rto.query_str('CHANnel2:DATA:VAL?')  # some non ascii
            ↪  bytes causing troubles
15          rto.query_opc()
16
17          time_vec = np.arange(header[0] * 1e9, header[1] * 1e9,
            ↪  ((header[1]-header[0]) * 1e9)/header[2])   # Times e9 to get
            ↪  rid of rounding errors
18          time_vec = np.around(time_vec, decimals=3)/1e9
            ↪  #round to 1ps and scale back
19          #np.set_printoptions(threshold=np.inf)
20          #print(time_vec)
21
22          data_ch1 = np.fromstring(data_ch1, dtype=float, sep=',')
23          data_ch1 = data_ch1*2.02
24          data_ch2 = np.fromstring(data_ch2, dtype=float, sep=',')
25          data_ch2 = data_ch2 / CH2_probe_factor
26
27          #Average data to get rid of noise,
28          scope_res = float(rto.query_str("ACQ:RES?"))
29          window = 6e-9   #6ns averaging window
30          window_samples = int(window/scope_res)
31          data_ch1_av_filt = np.convolve(data_ch1, np.ones(window_samples)
            ↪  / window_samples, mode='same')
```

```
32          data_ch2_av_filt = np.convolve(data_ch2, np.ones(window_samples)
            ↪  / window_samples, mode='same')
33
34          #Find offset of output voltage
35          index_ch1_offset = int(np.where(time_vec == -20e-9)[0])
36          ch1_offset = data_ch1_av_filt[index_ch1_offset]
37
38          #Find offset of input voltage
39          index_ch2_offset = int(np.where(time_vec == -20e-9)[0])
40          ch2_offset = data_ch2_av_filt[index_ch2_offset]
41
42          #Filter CH2 for finding peak
43          b, a = signal.butter(3, 1e9, 'low', fs=1/scope_res)
44          ch2_butter_filt = signal.filtfilt(b, a, data_ch2)
45
46          #Save max of CH2 and CH1 value after measurement delay
47          CH2_max = max(ch2_butter_filt) - ch2_offset
48          index_measure_delay = int(np.where(time_vec ==
            ↪  measure_delay)[0])
49          CH1_value = data_ch1_av_filt[index_measure_delay] - ch1_offset
50
51          rto.write_str("RUN")
52
53          return CH2_max, CH1_value
```

For long-time analysis the data readout is very similar, with the only difference, that the output voltage is measured after different time steps.

```
1       def RTO_save_CH1_and_CH2_csv_longtime(filename, CH2_probe_factor):
2           rto.write_str("STOP;")
3           rto.query_opc()
4           header_str_ch2 = rto.query_str('CHANnel1:DATA:HEADer?')      #
            ↪  Header sometimes makes problems restart scope
5           rto.query_opc()
6
7           header = np.fromstring(header_str_ch2, dtype=float, sep=',')
8
9           #rto.write_str("EXPort:WAVeform:INCXvalues OFF")      #
            ↪  interleaved xy on/off
10          data_ch1 = rto.query_str('CHANnel1:DATA:VAL?')  # some non ascii
            ↪  bytes causing troubles
11          rto.query_opc()
12          data_ch2 = rto.query_str('CHANnel2:DATA:VAL?')  # some non ascii
            ↪  bytes causing troubles
13          rto.query_opc()
14
```

```
15          time_vec = np.arange(header[0]* 1e9, header[1]* 1e9,
            ↪  ((header[1]-header[0]) * 1e9)/header[2])   # Times e8 to get
            ↪  rid of rounding errors
16          time_vec = np.around(time_vec, decimals=3)/1e9
            ↪  #round to 1ps and scale back
17
18
19          data_ch1 = np.fromstring(data_ch1, dtype=float, sep=',')
20          data_ch1 = data_ch1 * 2.02
21          data_ch2 = np.fromstring(data_ch2, dtype=float, sep=',')
22          data_ch2 = data_ch2 / CH2_probe_factor
23
24          #Average data to get rid of noise,
25          data_ch1_av_filt = np.convolve(data_ch1, np.ones(60) / 60,
            ↪  mode='same')
26          data_ch2_av_filt = np.convolve(data_ch2, np.ones(60) / 60,
            ↪  mode='same')
27
28          #Find offset of output voltage
29          index_ch1_offset = int(np.where(time_vec == -20e-9)[0])
30          ch1_offset = data_ch1_av_filt[index_ch1_offset]
31
32          #Find offset of input voltage
33          index_ch2_offset = int(np.where(time_vec == -20e-9)[0])
34          ch2_offset = data_ch2_av_filt[index_ch2_offset]
35
36          #Filter CH2 for finding peak
37          scope_res = float(rto.query_str("ACQ:RES?"))
38          b, a = signal.butter(3, 1e9, 'low', fs=1/scope_res)
39          ch2_butter_filt = signal.filtfilt(b, a, data_ch2)
40
41          data = np.column_stack((time_vec, data_ch1, data_ch2))    #add
            ↪  time vector to data
42          #np.savetxt(filename+".txt", data)
43
44          #Save max of CH2 and values for CH1 after 20ns, 80ns, 500ns,
            ↪  1us, 2us, 5us
45          CH2_max = max(ch2_butter_filt) - ch2_offset
46
47          index_30ns = int(np.where(time_vec == 30e-9)[0])
48          index_80ns = int(np.where(time_vec == 80e-9)[0])
49          index_200ns = int(np.where(time_vec == 200e-9)[0])
50          index_500ns = int(np.where(time_vec == 500e-9)[0])
51          index_1us = int(np.where(time_vec == 1e-6)[0])
52          index_3us = int(np.where(time_vec == 3e-6)[0])
53          index_5us = int(np.where(time_vec == 5e-6)[0])
```

```
54          index_10us = int(np.where(time_vec == 10e-6)[0])
55
56          CH1_vals = np.array([data_ch1[index_30ns], data_ch1[index_80ns],
            ↪  data_ch1[index_200ns], data_ch1[index_500ns],
            ↪  data_ch1[index_1us], data_ch1[index_3us],
            ↪  data_ch1[index_5us], data_ch1[index_10us]])
57          CH1_vals = CH1_vals - ch1_offset
58
59          rto.write_str("RUN")
60
61          return CH2_max, CH1_vals
```

D.6 Analysis Code

D.6.1 Long-Time Analysis

For the long-time analysis in PWL_long_time_analysis(...) the specifications of the test waveform are specified at the beginning. In a next step the amplitude is stepped in a loop by increasing the attenuation. The measurementdata afterwards is plotted and stored as text file.

```
1       def PWL_long_time_analysis(pulse_amp_max, amplitude,
        ↪  CH2_probe_factor, max_att, tail_current_mA):
2           print('Started PWL pulse long time analysis')
3
4           fs = 5e9
5           rise_time_ns = 4     #0.2ns steps for fs 5GS/s
6           on_time_ns = 10      #0.2ns steps for fs 5GS/s
7           fall_time_ns = 4     #0.2ns steps for fs 5GS/s
8
9           #Set circuit to op
10          set_circuit_op(tail_current_mA)
11
12          rto.write_str("ACQ:POIN:AUTO RESolution")  # Define Horizontal
            ↪  scale by number of points
13          rto.write_str("ACQ:RES 100e-12")  # 10ps Resolution
14          rto.write_str("TIM:RANG 0.000012")  # 10us Acquisition time
15
16          # Generate waveform
17          wfmData = trap_pulse(amplitude, rise_time_ns, on_time_ns,
            ↪  fall_time_ns, fs)
18          # Initalize AWG with defined waveform and sampling rate
19          AWG_init(wfmData, fs)
20
21          in_v_max = np.zeros(max_att+1)
22          out_v = np.zeros([max_att+1, 8])
```

```
23
24          for amp_count in range(0, max_att+1):
25              amp = pulse_amp_max * 10**(-amp_count/20)
26              RTO_set_vertical_scale(amp, CH2_probe_factor)    # Adjust
                ↪ vertical scale of scope
27              ATT_set(amp_count)
28              time.sleep(1)
29              AWG_trig()
30              time.sleep(0.5)
31              in_v_max[amp_count], out_v[amp_count, :] =
                ↪ RTO_save_CH1_and_CH2_csv_longtime(f"Waveform_PWL_long_
                ↪ time_att_{amp_count}dB_tr_{rise_time_ns}ns_on_{on_time_ns}
                ↪ ns_fall_{fall_time_ns}ns_{tail_current_mA}mA",
                ↪ CH2_probe_factor)
32
33
34
35          #Generate plots of transfere characteristic
36          #
37          out_v[out_v < 1e-4] = 1e-4 #Minimum value, important for
            ↪ logarithmic plots
38          for plot_count in range(0, 8):
39              plt.plot(in_v_max, out_v[:, plot_count])
40          plt.grid(True)
41          plt.legend(['after 30ns','after 80ns','after 200ns','after
            ↪ 500ns','after 1us','after 3us','after 5us','after 10us'])
42          plt.xlabel('Input voltage')
43          plt.ylabel('Output voltage')
44          plt.title(f'Long time behavior, tail current
            ↪ {tail_current_mA}mA')
45          plt.savefig(f'Longtime_behavior_10us_{tail_current_mA}mA.png',
            ↪ dpi=200, bbox_inches='tight')
46          plt.xlim((0, 0.1))
47
            ↪ plt.savefig(f'Longtime_behavior_10us_{tail_current_mA}mA_ss.png',
            ↪ dpi=200, bbox_inches='tight')
48          plt.close()
49
50          #Generate decay plots
51          time_decay = [30, 80, 200, 500, 1000, 3000, 5000, 10000]
52          for plot_count in range(0, len(time_decay)):
53              plt.plot(time_decay, out_v[plot_count, :])
54          plt.grid(True)
55          plt.legend(['0dB att','2dB att','5dB att','6dB att'])
56          plt.xlabel('Time in ns')
57          plt.ylabel('Output voltage')
```

```
58          plt.title(f'Long time behavior voltage droop, tail current
            ↪ {tail_current_mA}mA')
59
            ↪ plt.savefig(f'Longtime_10us_voltage_droop_{tail_current_mA}mA.png',
            ↪ dpi=200, bbox_inches='tight')
60          plt.close()
61
62
            ↪ np.savetxt(f'Long_time_results_inputv_max_{tail_current_mA}mA.txt',
            ↪ in_v_max)
63          np.savetxt(f'Long_time_results_outputv_after_different_times_
            ↪ {tail_current_mA}mA.txt', out_v)
64
            ↪ np.savetxt(f'Long_time_results_sample_times_{tail_current_mA}mA.txt',
            ↪ time_decay)
65
66          print('Completed PWL pulse long time analysis')
67
68          pass
```

D.6.2 Rise-Time Analysis

The PWL_pulse_risetime_analysis(...) function is very similar to the long-time analysis function, the main difference is that also the pulse shape is updated in a loop.

```
1       def PWL_pulse_risetime_analysis(pulse_amp_max, amplitude,
        ↪ CH2_probe_factor, max_att, tail_current_mA):
2           print('Started pulse rise time analysis')
3           fs = 10e9
4           rise_time_ns = [0.24, 0.72, 1.6, 2.4, 4]    #0.1ns steps for fs
            ↪ 10GS/s
5           # 0.24ns -> 0.8ns, 0.72ns -> 1ns,
6           on_time_ns = 10     #0.2ns steps for fs 5GS/s
7           fall_time_ns = 4    #0.2ns steps for fs 5GS/s
8
9           #Set circuit to op
10          set_circuit_op(tail_current_mA)
11
12          rto.write_str("ACQ:POIN:AUTO RESolution")  # Define Horizontal
            ↪ scale by number of points
13          rto.write_str("ACQ:RES 100e-12")  # 10ps Resolution
14          rto.write_str("TIM:RANG 0.000004")  # s Acquisition time
15
16          in_v_max = np.zeros([max_att+1, len(rise_time_ns)])
17          out_v = np.zeros([max_att+1, len(rise_time_ns)])
18
19          for pulse_r_count in range(0,len(rise_time_ns)):
```

```
20              #Generate waveform
21              wfmData = trap_pulse(amplitude, rise_time_ns[pulse_r_count],
                ↪  on_time_ns, fall_time_ns, fs)
22              #Initalize AWG with defined waveform and sampling rate
23              AWG_init(wfmData, fs)
24
25              for amp_count in range(0, max_att+1):
26                  amp = pulse_amp_max * 10**(-amp_count/20)
27                  RTO_set_vertical_scale(amp, CH2_probe_factor)    #Adjust
                    ↪  vertical scal of scope
28
29                  #att_req = pulse_amp_max/pulse_amp[amp_count]
30                  #att_req_db = 20 * np.log10(att_req)
31                  ATT_set(amp_count)
32                  time.sleep(1)
33                  AWG_trig()
34                  time.sleep(0.5)
35                  in_v_max[amp_count, pulse_r_count], out_v[amp_count,
                    ↪  pulse_r_count] =
                    ↪  RTO_read_CH1_max_and_CH2_after_ns(100,
                    ↪  CH2_probe_factor)
36
37          #Generate plots of transfere characteristic
38          out_v[out_v < 1e-4] = 1e-4 #Minimum value, important for
            ↪  logarithmic plots
39          for pulse_number in range(0, len(rise_time_ns)):
40              plt.plot(in_v_max[:, pulse_number], out_v[:, pulse_number])
41          plt.grid(True)
42          #Values from real rise-time measurement, in software defined
            ↪  riesetime for fast signals not 100% right
43          plt.legend(['rise-time 0.8ns', 'rise-time 1ns', 'rise-time
            ↪  1.6ns', 'rise-time 2.4ns', 'rise-time 4ns'])
44          plt.xlabel('Input voltage')
45          plt.ylabel('Output voltage')
46          plt.title(f'Pulse rise-time analysis, tail current
            ↪  {tail_current_mA}mA')
47          plt.savefig(f'Pulse_rise-time_analysis_{tail_current_mA}mA.png',
            ↪  dpi=250, bbox_inches='tight')
48          plt.xlim((0, 0.1))
49
            ↪  plt.savefig(f'Pulse_rise-time_analysis_{tail_current_mA}mA_ss.png',
            ↪  dpi=250, bbox_inches='tight')
50          plt.close()
51
52          #real rise-times with amplifier fot saving
53          rise_time_ns[0] = 0.8
```

```
54          rise_time_ns[1] = 1
55
            ↪ np.savetxt(f'rise_time_analysis_data_input_max_{tail_current_mA}mA.txt',
            ↪ in_v_max)
56          np.savetxt(f'rise_time_analysis_data_outptutv_after_1us_
            ↪ {tail_current_mA}mA.txt', out_v)
57
            ↪ np.savetxt(f'rise_time_analysis_data_rise_times_{tail_current_mA}mA.txt',
            ↪ rise_time_ns)
58
59          print('Completed pulse rise time analysis')
60          pass
```

D.6.3 Pulse-Width Analysis

```
1       def PWL_pulse_width_analysis(pulse_amp_max, amplitude,
        ↪ CH2_probe_factor, max_att, tail_current_mA):
2           print('Started pulse width analysis')
3           fs = 10e9
4           rise_time_ns = 0.24    #0.1ns steps for fs 10GS/s, 0.24 -> 0.8ns
            ↪ rt with amplifier
5           on_time_ns = [1.5, 2, 3, 4, 10]     #0.2ns steps for fs 5GS/s
6           fall_time_ns = 0.24    #0.1ns steps for fs 10GS/s
7
8           #Set circuit to op
9           set_circuit_op(tail_current_mA)
10
11          rto.write_str("ACQ:POIN:AUTO RESolution")  # Define Horizontal
            ↪ scale by number of points
12          rto.write_str("ACQ:RES 100e-12")  # 10ps Resolution
13          rto.write_str("TIM:RANG 0.000004")  # s Acquisition time
14
15          in_v_max = np.zeros([max_att+1, len(on_time_ns)])
16          out_v = np.zeros([max_att+1, len(on_time_ns)])
17
18
19          for pulse_w_count in range(0,len(on_time_ns)):
20              #Generate waveform
21              wfmData = trap_pulse(amplitude, rise_time_ns,
                ↪ on_time_ns[pulse_w_count], fall_time_ns, fs)
22              #Initalize AWG with defined waveform and sampling rate
23              AWG_init(wfmData, fs)
24
25              for amp_count in range(0, max_att+1):
26                  amp = pulse_amp_max * 10**(-(amp_count)/20)
```

```
27                  RTO_set_vertical_scale(amp, CH2_probe_factor)    #Adjust
                    ↪  vertical scal of scope
28
29                  #att_req = pulse_amp_max/pulse_amp[amp_count]
30                  #att_req_db = 20 * np.log10(att_req)
31                  ATT_set(amp_count)
32                  time.sleep(1)
33                  AWG_trig()
34                  time.sleep(0.5)
35                  in_v_max[amp_count, pulse_w_count], out_v[amp_count,
                    ↪  pulse_w_count] =
                    ↪  RTO_read_CH1_max_and_CH2_after_ns(100,
                    ↪  CH2_probe_factor)
36
37
38          #Generate plots of transfere characteristic
39          #
40          out_v[out_v < 1e-4] = 1e-4 #Minimum value, important for
            ↪  logarithmic plots
41          for pulse_number in range(0, len(on_time_ns)):
42              plt.plot(in_v_max[:, pulse_number], out_v[:, pulse_number])
43          plt.grid(True)
44          plt.legend(['Duration 1.5ns','Duration 2ns','Duration
            ↪  3ns','Duration 4ns', 'Duration 10ns'])
45          plt.xlabel('Input voltage')
46          plt.ylabel('Output voltage')
47          plt.title(f'Pulse width analysis, tail current
            ↪  {tail_current_mA}mA')
48          plt.savefig(f'Pulse_width_analysis_{tail_current_mA}mA.png',
            ↪  dpi=200, bbox_inches='tight')
49          plt.xlim((0, 0.1))
50          plt.savefig(f'Pulse_width_analysis_{tail_current_mA}mA_ss.png',
            ↪  dpi=200, bbox_inches='tight')
51          plt.close()
52
53          np.savetxt(f'Pulse_width_analysis_results_inputv_max_
            ↪  {tail_current_mA}mA.txt', in_v_max)
54          np.savetxt(f'Pulse_width_analysis_outputv_after_1us_times_
            ↪  {tail_current_mA}mA.txt', out_v)
55
            ↪  np.savetxt(f'Pulse_width_analysis_ontimes_{tail_current_mA}mA.txt',
            ↪  on_time_ns)
56
57          print('Completed pulse width analysis')
58
59          pass
```

D.7 Main

In the main, all analysis functions are started and the operating point is changed in a loop.

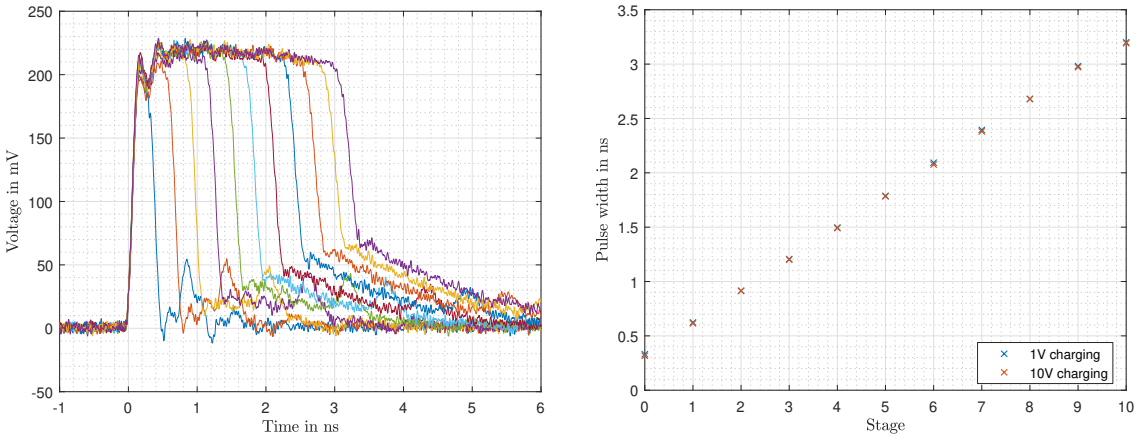
```
    def main():
        print('Operation started \nYou can go for a coffee, this my take a while!')

        pulse_amp_max = 3.9      #3.5V with step att_a -> -20dB -> -6dB -> +40dB -> Step att_b
        amplitude = -0.5         #Negative for setup with inveting Amplifier!!!
        CH2_probe = 51           #Value of probing resistor
        max_attenuation = 53     #15dB step att + 32dB step att + plus AWG amplitude reduction

        CH2_probe_factor = 1/(CH2_probe/50 + 1)

        RTO_initialization()

        #Connect to programmeble attenuator and set communication parameters
        ATT_init()

        time_domain_analysis(pulse_amp_max, amplitude, CH2_probe_factor, 15)
        tail_current_mA = 15

        tail_current_values_mA = [5, 10, 15, 20, 25]
        for measurement in range(0, 5):
            tail_current_mA = tail_current_values_mA[measurement]
            PWL_long_time_analysis(pulse_amp_max, amplitude, CH2_probe_factor, max_attenuation, tail_current_mA)
            PWL_pulse_risetime_analysis(pulse_amp_max, amplitude, CH2_probe_factor, max_attenuation, tail_current_mA)
            PWL_pulse_width_analysis(pulse_amp_max, amplitude, CH2_probe_factor, max_attenuation, tail_current_mA)

        # Close the session
        rto.close()
        print('Completed, be happy with your results \nEverything closed!')

        pass

    start_time = time.time()
```

```
33      main()
34      print("--- %s minutes ---" % ((time.time() - start_time)/60))
```

Appendix E

Variable Transmission-Line-Pulse Generator

For fast initial measurements a transmission line pulse (TLP) generator was designed with adjustable pulse-width. It supports pulse-widths from about 330ps to 3.3ns in 300ps steps. The basic principle of the TLP is that a transmission line is charged by a high ohmic resistor and by triggering a mercury relay with a magnet a pulse is generated [20]. The pulse-width is determined by the length of the transmission line. For adjusting the pulse-width rf-relay were used to partially extend the length of the transmission line. For controlling the relays a shift-register with open drain outputs was used, which is controlled by a button. The full schematic can be obtained from figure E.3 and E.4. In figure E.2 the top view of the PCB is shown without the mercury relay. For reducing impedance mismatch at the mercury relay the relay was wrapped in copper-tape and soldered to the ground plane underneath.



(a) Time domain result 1V charge voltage

(b) Pulse-Width for 1V and 10V charge voltage

Figure E.1: TLP-generator Measurement Result

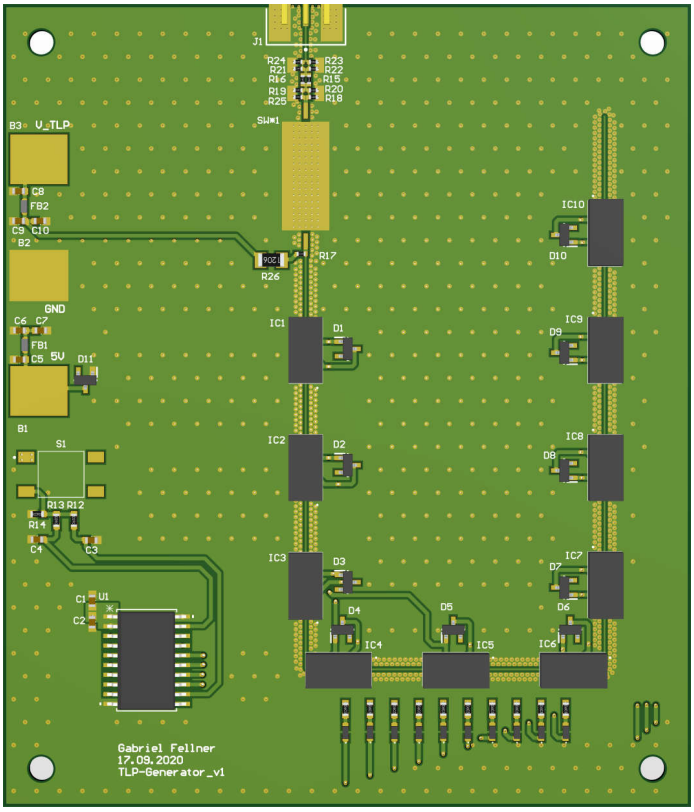


Figure E.2: PDH using bipolar Transistors PCB

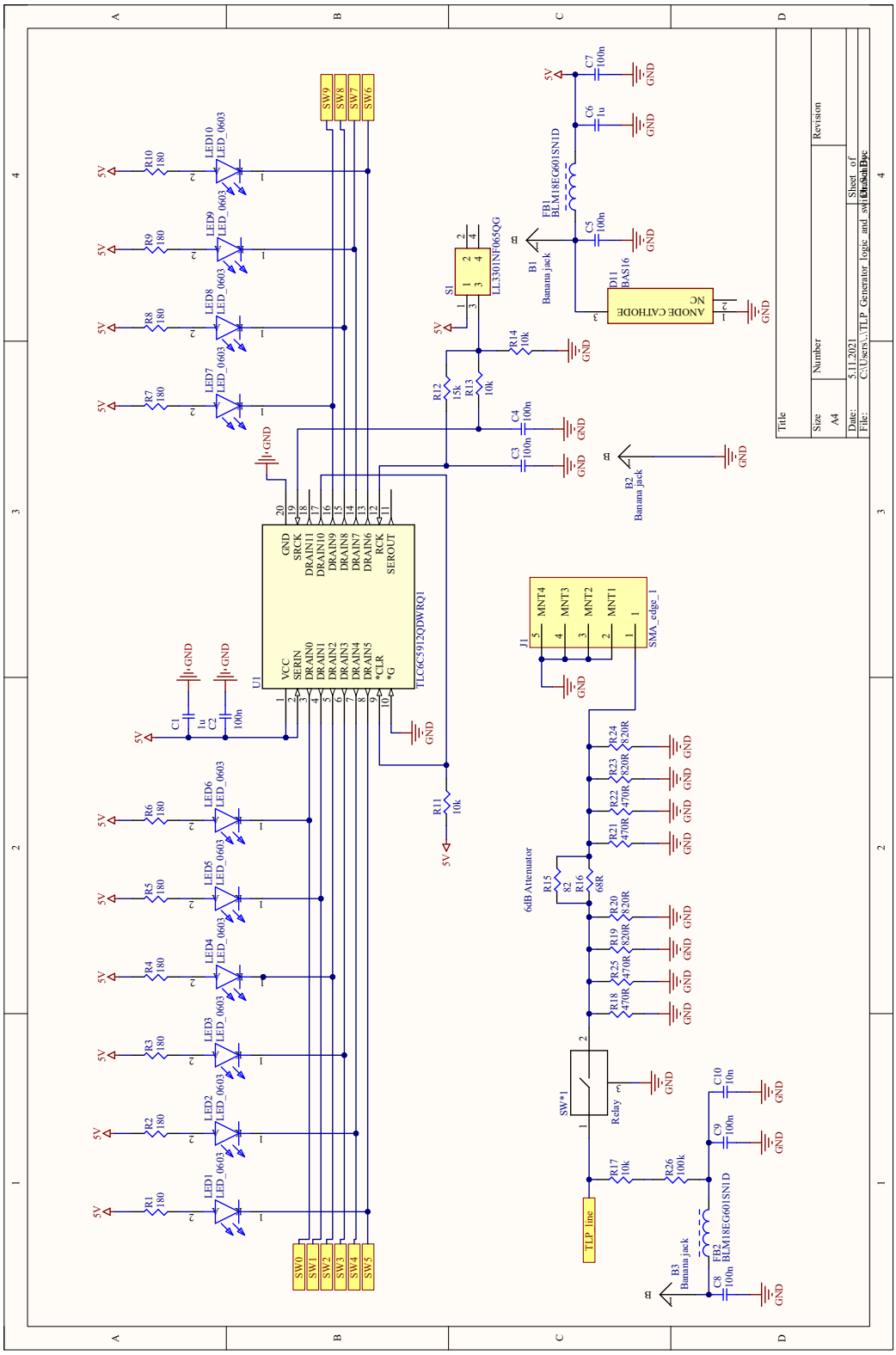


Figure E.3: PDH using bipolar Transistors Schematic Page 1

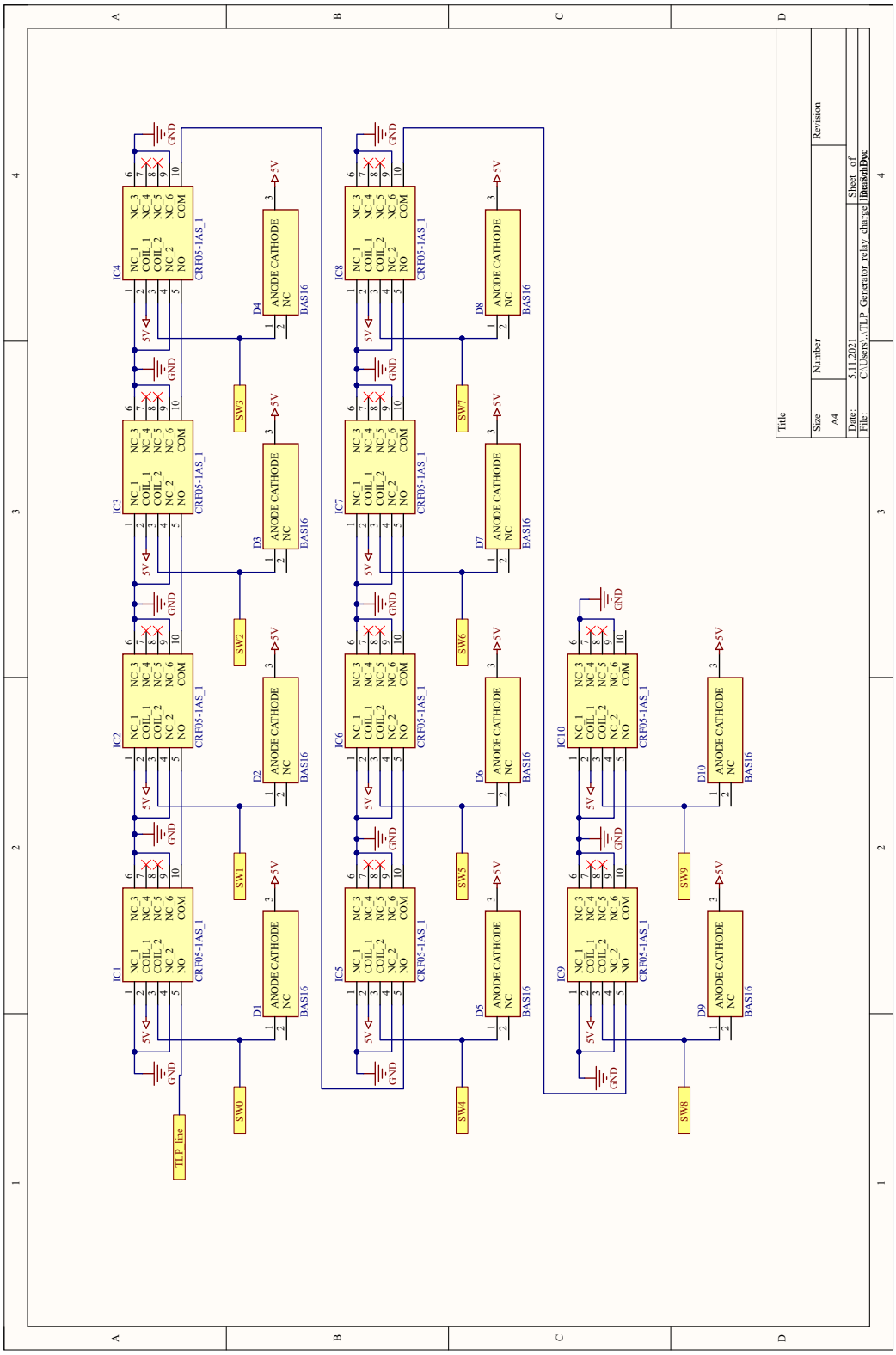


Figure E.4: PDH using bipolar Transistors Schematic Page 2

Bibliography

[1] R. B. Simonic, "Electrostatic furniture discharge event rates for metallic-covered, floor-standing information processing machines," in *1982 IEEE International Symposium on Electromagnetic Compatibility*, 1982, pp. 1–8.

[2] S. Frei and D. Pommerenke, "A transient field measurement system to analyze the severity and occurrence rate of electrostatic discharge (esd)," *Journal of Electrostatics*, vol. 44, no. 3, pp. 191–203, 1998. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0304388698000382

[3] P. Horowitz, *The art of electronics*. Cambridge England New York: Cambridge University Press, 1989.

[4] K. Achtenberg, J. Mikolajczyk, D. Szabra, A. Prokopiuk, and Z. Bielecki, "Review of peak signal detection methods in nanosecond pulses monitoring," *Metrology and Measurement Systems*, vol. 27, pp. 203–218, 06 2020.

[5] K. Gopalan, R. A. Roberts, J. G. Markovich, and J. J. Vaitekunas, "A high-speed digital peak detector and averager for acoustic microscopy," in *6th IEEE Conference Record., Instrumentation and Measurement Technology Conference*, 1989, pp. 54–56.

[6] M. W. Kruiskamp and D. M. W. Leenaerts, "A cmos peak detect sample and hold circuit," *IEEE Transactions on Nuclear Science*, vol. 41, no. 1, pp. 295–298, 1994.

[7] G. Geronimo, P. Connor, and A. Kandasamy, "Analog cmos peak detect and hold circuits. part 1. analysis of the classical configuration* 1," *Nuclear Instruments and Methods in Physics Research A*, vol. 4845030, pp. 533–543, 05 2002.

[8] G. Geronimo, P. O'Connor, and A. Kandasamy, "Analog cmos peak detect and hold circuits. part 2. the two-phase offset-free and derandomizing configuration," *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, vol. 484, pp. 544–556, 05 2002.

[9] K. Achtenberg, J. Mikolajczyk, D. Szabra, A. Prokopiuk, and Z. Bielecki, "Optical pulse monitoring unit for free space optics," *Opto-Electronics Review*, vol. 27, pp. 291–297, 09 2019.

[10] W. Zhang, H. Zhang, X. Zhang, and Y. Chen, "Laser pulse peak holding circuit for low cost laser tracking applications," 10 2016, p. 1015306.

[11] W. John, "Peak detectors gain in speed and performance – design note 61," Tech. Rep., 1992. [Online]. Available: https://www.analog.com/media/en/reference-design-documentation/design-notes/dn61f.pdf

[12] A. Gabino and K. Hassan. (2016) Ltc6244 high speed peak detector. [Online]. Available: https://www.analog.com/en/technical-articles/ltc6244-high-speed-peak-detector.html

[13] W. Haas and P. Dullenkopf, "A novel peak amplitude and time detector for narrow pulse signals," *IEEE Transactions on Instrumentation and Measurement*, vol. IM-35, no. 4, pp. 547–550, 1986.

[14] D. Costin and P. Opris, "High speed peak detector for glitch-catching used in digital storage scopes," in *1995 International Semiconductor Conference. CAS '95 Proceedings*, 1995, pp. 233–236.

[15] R. N. Larsen, "Nanosecond pulse stretcher," *Review of Scientific Instruments*, vol. 37, no. 4, pp. 514–515, 1966. [Online]. Available: https://doi.org/10.1063/1.1720228

[16] P. Buckens and M. Veatch, "A high performance peak-detect and hold circuit for pulse height analysis," *IEEE Transactions on Nuclear Science*, vol. 39, no. 4, pp. 753–757, 1992.

[17] P. Y. Chang and H. P. Chou, "A high precision peak detect sample and hold circuit," in *2006 IEEE Nuclear Science Symposium Conference Record*, vol. 1, 2006, pp. 329–331.

[18] L. D. Tekumala, "On-chip characterization of single-event charge-collection," Master's thesis, Graduate School of Vanderbilt University, The address of the publisher, 8 2012.

[19] Chen, *Analog circuits and devices.* Boca Raton, FL: CRC Press, 2003.

[20] T. Maloney and N. Khurana, "Transmission line pulse technique for circuit modeling and esd phenomena," *Transmission line pulsing techniques for circuit modeling of ESD phenomena*, pp. 49–54, 01 1985.