ACKNOWLEDGEMENTS

I would like to thank Dr. George Engel, for his encouragement, support and guidance and also having given me an opportunity to work with him as his research assistant. He has been a positive factor in my academic experience here at Southern Illinois University, Edwardsville.

I would like to thank Dr. Lee Sobotka and Mr. Jon Elson, department of chemistry, Washington University Saint Louis, for their help during the various stages of this project. My special thanks to the faculty and staff of ECE department for their direct and indirect support without which I simply could not have progressed with my work.

I am extremely grateful to Mythreyi Nethi and Jon C. Wade who were of great help during the entire course of my thesis work. I would also like to thank Balaji Golla, Arun Gerard and all my other friends for their constant support during my entire studies at SIUE.

This acknowledgement would not be complete without a word of appreciation to my parents and brother who have been of constant encouragement all my life. Without their support I would not have come so far in my life.

TABLE OF CONTENTS

LIST OF FIGURES

Figure

LIST OF TABLES

Table

CHAPTER 1

INTRODUCTION

1.1 Research Background

The nucleus, core of an atom, contains most of its mass. The nucleus is positively charged and contains one or more nucleons (protons or neutrons). The positive charge of the nucleus is determined by the number of protons it contains; in the neutral atom this positive charge is balanced by an equal number of negatively charged electrons orbiting the nucleus in a comparatively large region outside it.

Nuclei came into being through cataclysmic events in the evolution of the universe: light nuclei were created within a few minutes after the Big Bang, while the heavier nuclei were, and even to this day continue to be produced in stars, the cauldrons of the cosmos. In our quest to understand the fundamentals underlying the physical world that we live in, and the origins of its formation and evolution, it is essential to have a full understanding of the atomic nucleus. The process of heavy elemental synthesis involves unstable nuclei at every stage. Therefore to understand the origin of elements that surround us and that are critical to both material and life processes, we need to study unstable nuclei.

Nuclei come in a wide variety of combinations of protons and neutrons. However, due to the underlying forces and symmetries of the laws of Nature, not all combinations are possible. As a result stable nuclei occupy only a narrow band of the proton number verses neutron number space of the chart of the nuclides. Light nuclei tend to have equal numbers of neutrons and protons and as the atomic number (proton number) increases the number of neutrons needed for stability grows at a rate slightly faster than that for protons. As a result

at the upper end of the periodic table there are approximately 50% more neutrons than protons in stable nuclei.

Nuclei, off this narrow band, beta decay back to stability. Beta decay however is moderated by the weak interaction and therefore the lifetimes for beta-unstable (but strong interaction stable) can be quite long. In fact beta decay lifetimes are always greater than 1 millisecond.

With very few exceptions, for fifty years the study of the reactions between atomic nuclei has been confined to reactions, which can be studied with beta-stable target and projectiles. Recently accelerator techniques have been developed which provide beams of (beta) unstable nuclei. Two techniques can be used to do this. One involves using two accelerators with a thick stopping ``production target'' in between. The beta unstable nuclei are produced by nuclear reactions in the production target. These nuclei are extracted from the target, injected, and accelerated by the second machine and delivered to a second target. The second method for generating beams of beta unstable nuclei uses the fragmentation technique in which high-energy primary beams are fragmented on a thin production target and a beta unstable reaction product is selected with a magnetic analysis system and delivered to a secondary target. The first technique generally produces more intense secondary beams but can only be used to generate beams of nuclei with lifetimes greater than about 1 s. The second technique can be used to generate beams of any beta unstable nucleus (which can be produced by fragmentation) as the magnetic analysis system refocuses the secondary beam in less than a microsecond. A major fragmentation facility presently exists in Germany. Upgrades to facilities in France and at the National Super conducting Cyclotron

Laboratory in East Lansing Michigan are underway. A major initiative to build a facility of the first type is being planned at Argonne National Laboratory.

There are several reasons to study reactions between beta unstable nuclei. While the nuclei, which compose all the materials on earth, are beta-stable (half-lives on the order of, or longer than, the age of the universe) these nuclei were made by beta-unstable synthetic pathways. For example most nuclei with masses greater than 56 were made in supernova explosions via a pathway (of rapid neutron capture), which passes though nuclei far out into the beta unstable region. (In this case with great neutron excess.) After the stellar explosion, the nuclei beta-decay back to stability. Prior to the availability of beta unstable beams these nucleosynthesis pathways could not be studied.

The fragmentation method of producing secondary beams is in fact itself similar to another astrophysical process. Cosmic rays produce unstable nuclei when they impinge on our atmosphere. The creation of 14C is the best-known case of this. The new experimental techniques allow us to study reactions with 14C and for that matter, beams of 9C-18C. The ability to vary the neutron to proton content of the reaction system allows us to study, in an unprecedented fashion, the strength of the nuclear force in the limits of the n/p degree of freedom. Remember that while nuclei like 14C are beta (weak interaction) unstable they are well bound by the strong force. (This means they do not decay in a fashion, which changes the atomic mass number, but rather change the neutrons into protons. The process is the slow beta decay process mentioned above.) The new facilities thus allow us to engage in reaction studies using many more strongly bound (weakly unbound) systems.

1.2 Need for an Integrated Circuit

The primary difficulty in performing experiments at these new facilities using secondary beams is that the beam intensities are many orders of magnitude lower than those of the primary beams. It is not uncommon to use primary beam intensities of 10^9 particles/s. Secondary beam intensities are never greater than 10^7 and some of the more interesting cases will involve beams of no more than 10^3 particles/s. This great loss in beam intensity must be offset in significant measure by an increase in the detection efficiency of the ultimate experiment. This has lead to very large solid angle detectors (sometimes 4pi) being built or designed. One such project is being designed by Washington University - Michigan State University – Indiana University collaboration. This project is unique in that is will make use of a very large array of solid-state Si detectors diode detectors. These detectors provide better energy resolution than the alternatives.

However in order for such a device to be useful, not only must the solid angle be large, but the position (angular) resolution must also be excellent. This means that the Si detector system must not only be large, but it also must contain many elements. In this case approximately 4000 elements will ultimately be needed. The charge generated in these reverse biased Si diode detector elements, when ionizing radiation passes though them, must be collected, amplified and digitized. Doing this with discrete electronics would be very expensive and too cumbersome to debug in a reasonable time. By designing a multi-channel CMOS chip with each channel containing a preamplifier, a pulse shaper, a discriminator, and pulse sequencers for the charge and timing outputs, one will be able to process the information from such a large number of diodes.

1.3 Features of HINP16C

HINP16C stands for Heavy Ion Nuclear Physics 16 Channels. This is used for high density signal processing in low and intermediate nuclear physics experiments. HINP16C possesses some features that no other commercial chip designed for these kinds of experiments posses. Some highlights of HINP16C are

1. It has two different gain modes, either 100Mev or 500Mev.
2. Built in high quality timing circuitry.
3. Self-triggering capability.
4. Capable of processing either polarity.
5. Capability to utilize external preamplifiers.
6. Data sparsification where the user can select the channels of his choice to be read out.

1.4 Sample Applications

The HINP16C can be used in a wide range of applications in the field of nuclear physics. Just a few sample applications are being described.

1. Spectroscopy of low-lying particle unstable states by resonance decay correlation techniques.
2. Direct and inelastic scattering of secondary unstable beams on p or d targets to study single-particle structure and excitation energy dependence of the nuclear level density.
3. Particle-particle correlation experiments at intermediate energy designed to refine source-size characteristics.

4. With external high gain and perhaps cooled preamplifiers, experiments requiring large arrays for detecting e^-'s and γ's.
5. Pixelated Si arrays for the ends caps of b-asymmetry measurements aimed at improving the error bounds on the CKM matrix for n decay.

1.5 Previous Work

There have been two integrated circuits that has been designed and fabricated as a part of this research work. The first IC that was fabricated had 8 channels, with each channel consisting of a CSA and a pulse shaper. Folded cascode amplifiers were used in building the pulse shaper. On testing the chip it was found that the system was not linear up to 15 million electrons and it was the shaper that limited the linearity of the system. Also the input referred noise was found to be more than 3000 electrons which was way beyond the expected value. The CSA and its bias generator were responsible for the noise sources within the system. For the next submission of the chip the pulse shaper was built using linear conductors to improve the linearity of the system and the CSA and its bias generator were also redesigned to give the required noise performance. The second IC had four channels, with each channel consisting of a CSA, a pulse shaper and a discriminator. On testing the chip it was found that the CSA and the pulse shaper worked extremely well, but the discriminator did not. As a result, the discriminator was redesigned with added features and the peak sampling circuit, time to voltage converter circuit, reset logic circuit, bias circuits, and common digital circuits were also added in the third IC. The rest of the report deals only with third IC that has 16 channels.

1.6 Object and Scope of This Thesis Work

The object of this thesis work is to design an integrated circuit that can identify radioactive elements by processing the charge emitted by the impacted silicon detectors. The integrated circuit consists of charge sensitive amplifier (CSA), pulse shaper, pseudo constant fraction discriminator (CFD), peak sampler, time to voltage converter (TVC), reset logic circuits, bias generator circuits and the common digital circuits. The thesis consists of 5 chapters. The design specifications and methodologies, the radiation sensor design, the subsystem design aspects providing details regarding the design and functionality of the individual blocks of the entire system are presented in Chapter 2.Simulation and experimental results are discussed in Chapter 3. Chapter 4 compares the simulated output with the actual output obtained from the chip. Chapter 5 provides the conclusions and the future direction of this research work. The IC has been implemented in 0.5 micron double poly, triple metal CMOS technology.

CHAPTER 2

HINP16C DESIGN

2.1 Radiation Sensor

The elementary particles and radiation events in particle physics and radiochemistry applications are detected by means of silicon detectors. For this purpose two silicon detectors that are 65 µm and 1.5 mm thick are used. The 65 µm detector is single sided with silicon strips in the vertical direction while the 1.5 mm detector is double sided with silicon strips that are vertical in the front and horizontal at the back.

The silicon detector is reverse biased photosensitive diode which detects charged particles or radiation events by generating electron-hole pairs in the detector material. The electrons thus generated tend to drift towards the positively biased n^+ contact of the detector and are finally collected. Since the time for collection is very small, of the order of nanoseconds, the detector output system can be represented as a current pulse, the integral of which is the total generated charge, Q.

2.2 System Specifications

There are certain requirements the radiation sensor should meet. The entire system should be linear to at least 80 Mev. The noise of the system referred back to the input should be less than 3000 electrons. The CSA should support two gain modes: 100 MeV and 500 Mev full-scale. The system should be able to use any external preamplifiers instead of the CSA. The discriminator should be present in the same IC as the CSA and the shaper. The

timing error of the discriminator should be less than ±500 ps. It should possess the ability to examine the internal analog output signals in real time.

2.3 System Level Design

The system level design of HINP16C is shown in Figure 2.1. It consists of three major blocks of circuits: linear circuits, timing circuits and the read out and control circuits. The CSA is the input stage for all the channels in the IC. The output of the CSA is split to feed the energy and timing branches each of which produce sparsified pulse trains with synchronized addresses for off-chip digitization with a pipelined ADC.

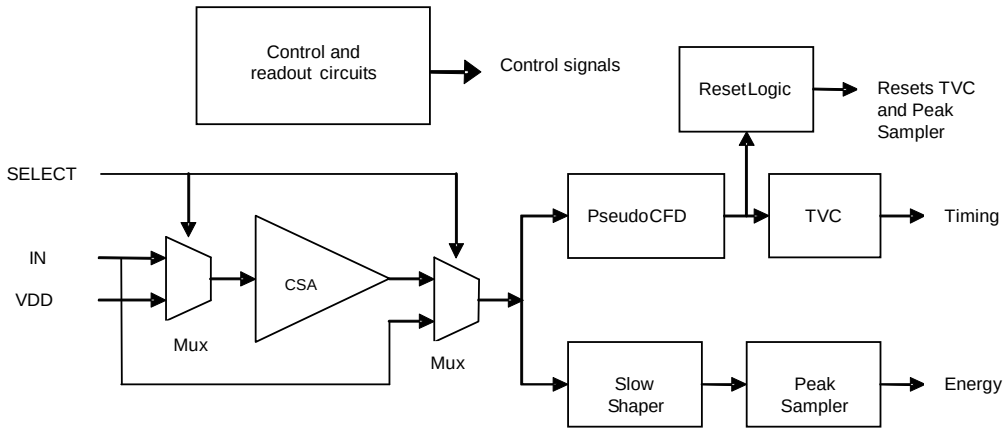


Figure 2.1 Different blocks of a single channel

2.3.1 Linear circuits

The linear circuits in the IC measure the amplitude (energy) of the charged particle at the input of the CSA. The shaper and the peak sampler are responsible for measuring the energy of the input signals. The exponentially decaying input to the shaper is converted into a Gaussian pulse. The voltage pulse at the output of the shaper is proportional to the energy of

the detected particle. This voltage output from the shaper is amplified and the peaking voltage of the Gaussian pulse stored by the peak sampler circuit.

2.3.2 Timing circuits

The precise measurement of time interval between the collision and the arrival of the signal at the output regardless of their intensity is done by the timing circuits in the IC. This system consists of a constant fraction discriminator and a time to voltage converter. The output of the CSA is fed to the CFD and it processes the signal and produces a logic high output indicating the channel is been hit. This signal is used to charge the capacitor present in the time to voltage converter. The amount of charge stored in the capacitor is proportional to the time of impact of the charged particle at the input of the CSA.

2.3.3 Control and readout circuits

The HINP16C consists of 48-bit configuration register that can be selectively loaded to produce various control signals for the proper functioning for the IC. The output of the configuration register can disable CFD outputs on a channel-by-channel basis, select test modes, select processing for either positive or negative CSA pulses, select CSA gain mode, TVC measurement range, and assign an 8-bit ID to the chip. The chip only responds when an externally applied chip address matches the ID stored in the chip's configuration register. Apart from the digital circuits, there are also analog circuits that provide proper biasing for the 16 processing channels.

The IC also accommodates some readout circuits that present useful processed data to the outside world. This includes the information on how many channels are hit, is any of the

channels are hit, an acknowledgement to indicate the completion of the acquisition process and the address of the channel currently being processed.

2.4 Charge Sensitive Amplifier (CSA)

2.4.1 Design specifications of the CSA

The design specifications of the CSA were the following.

(1) The value for the detector capacitance(C_{det}) should be in the range of 30pF and 150pF,

(2) A minimum sized charge packet of $4.806X10^{-16}$ C (3000 equivalent electrons) and a maximum charge packet of $2.403X10^{-12}$ C (15 million equivalent electrons) in the high gain mode and a maximum charge packet of $12.015X10^{-12}$ C (75 million equivalent electrons) in the low gain mode,

(3) an open loop gain of at least 80 dB,

(4) a phase margin of at least 45 degrees,

(5) input referred noise less than 3000 electrons,

(6) a 10-90 rise time , less than 75 ns,

(7) a decay time of 100 µs,

(8) a 5 V power supply and

(9) low power dissipation.

The CSA was designed to meet the above mentioned specifications. The design of the CSA will be explained in the following section. The transient and the noise performances of the CSA will be discussed in chapter 4.

2.4.2 Design of the CSA

The CSA shown in Figure 2.2 was designed to have two different gain settings. This was accomplished using an operational transconductance amplifier (OTA) and a bunch of switches. When operating in the high gain mode the values of the feedback capacitor is 2.5 pF and that of the feedback resistor is 10 MΩ. In this mode the maximum size of the charge packet that it can process is $2.403X10^{-12}$ C (15 million equivalent electrons). In the low gain mode the value of the feedback capacitor is 12.5 pF and the resistor is 2 MΩ and also it can process charge packets of sizes up to$12.015X10^{-12}$ C (75 million equivalent electrons).

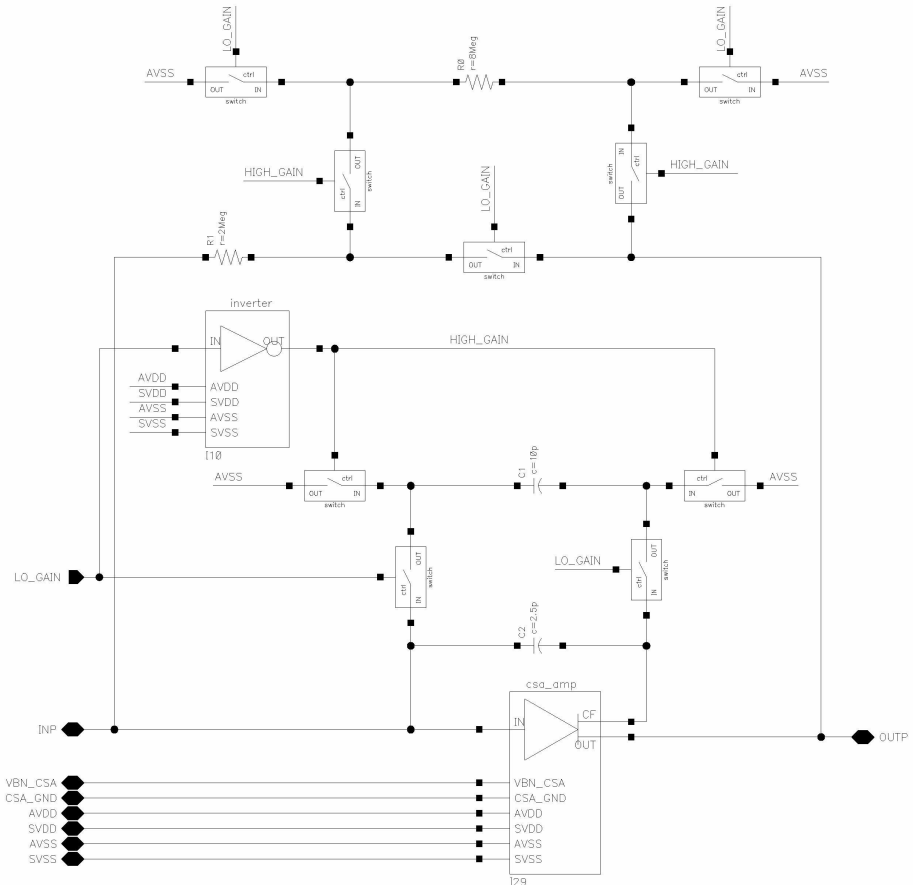


Figure 2.2 Block diagram of the CSA

To study the transient nature of the CSA, its transfer function $V_{out}(s)/ I_{in}(s)$ should be calculated. The closed loop transfer function of the system in the frequency domain is given by,

$$\frac{V_{out}(s)}{I_{in}(s)} = -\frac{g_m}{g_m G_f + s g_m C_f + s^2 C_t (C_f + C_L)} \tag{2.1}$$

where g_m is the transconductance of the core amplifier, C_L is the load capacitance at the output of the CSA, C_f is the feed back capacitance (C_1 and C_2 in Figure 2.3) and C_t is the combination of the capacitance at the input of the CSA, the detector capacitance and the C_f. It can be observed from equation (2.1) that the CSA is inverting in nature.

Assuming the poles are widely separated, the pole positions are given by ,

$$p_1 = \frac{1}{2\pi\tau_1} = \frac{1}{2\pi R_f C_f} \tag{2.2}$$

$$p_2 = \frac{1}{2\pi\tau_2} = \frac{g_m C_f}{2\pi C_t (C_L + C_f)} = \frac{GBWC_f}{C_t} \tag{2.3}$$

where GBW is the gain bandwidth product of the OTA.

The time constant $\tau_1 = R_f C_f$ determines the reset time or decay time of the CSA. The time constant, τ_2, determines the rise time of the CSA which is given by,

$$t_r = 2.2\tau_2 = 2.2\frac{C_t}{2\pi GBWC_f} \tag{2.4}$$

It can be seen from the expression that rise time t_r, is inversely proportional to the *GBW* of the CSA. The preamplifier must be sufficiently fast to integrate the charge *Q* on to the feedback capacitor C_f. This is an important requirement for detector readout systems where a high counting rate and short peaking time τ_s, are needed. As shown from the equation a high *GBW* for the core OTA can help in achieving a fast rise time for the CSA.

The maximum *GBW* or the minimum rise time is limited by the stability of the system which requires that all non-dominant poles must lie beyond p_2 i.e., beyond the unity gain frequency of the system. To reduce the noise contribution of R_f, its value should be made as high as possible. This would increase the reset time, which is in contradiction to the high counting rate requirement where a quick recovery of the output of the CSA is needed. Therefore a compromise is needed between the extra noise contribution of the feedback resistor and the counting rate requirements with regards to the value of the feedback resistor.

The OTA used in the CSA is a single ended, folded cascode circuit and is buffered by an n-source follower as shown by the schematic in Figure 2.3. It should be noted that the width and the length of the devices shown in the schematics are given in micrometers (µm).

The CSA configuration and sizing of the input transistor and the transistors in the bias circuit plays a very important role in the noise performance of the overall system. The following observations can be made from the analysis shown in [1].

(1) Equivalent noise charge due to thermal noise can be reduced by increasing the g_m of the input transistor.

(2) The thermal noise can be brought down by decreasing the length and increasing the drain to source current (I_{DS}) of the input transistor.

(3) The minimum value of the equivalent noise charge due to *1/f* noise is independent of the design parameters and depends only on the process parameters of the detector capacitance.

From the observations the width and the length of the input transistor were chosen to be 2457.6 µm and 0.9 µm respectively. A noise analysis was done to verify that the input referred noise was less than 3000 electrons.

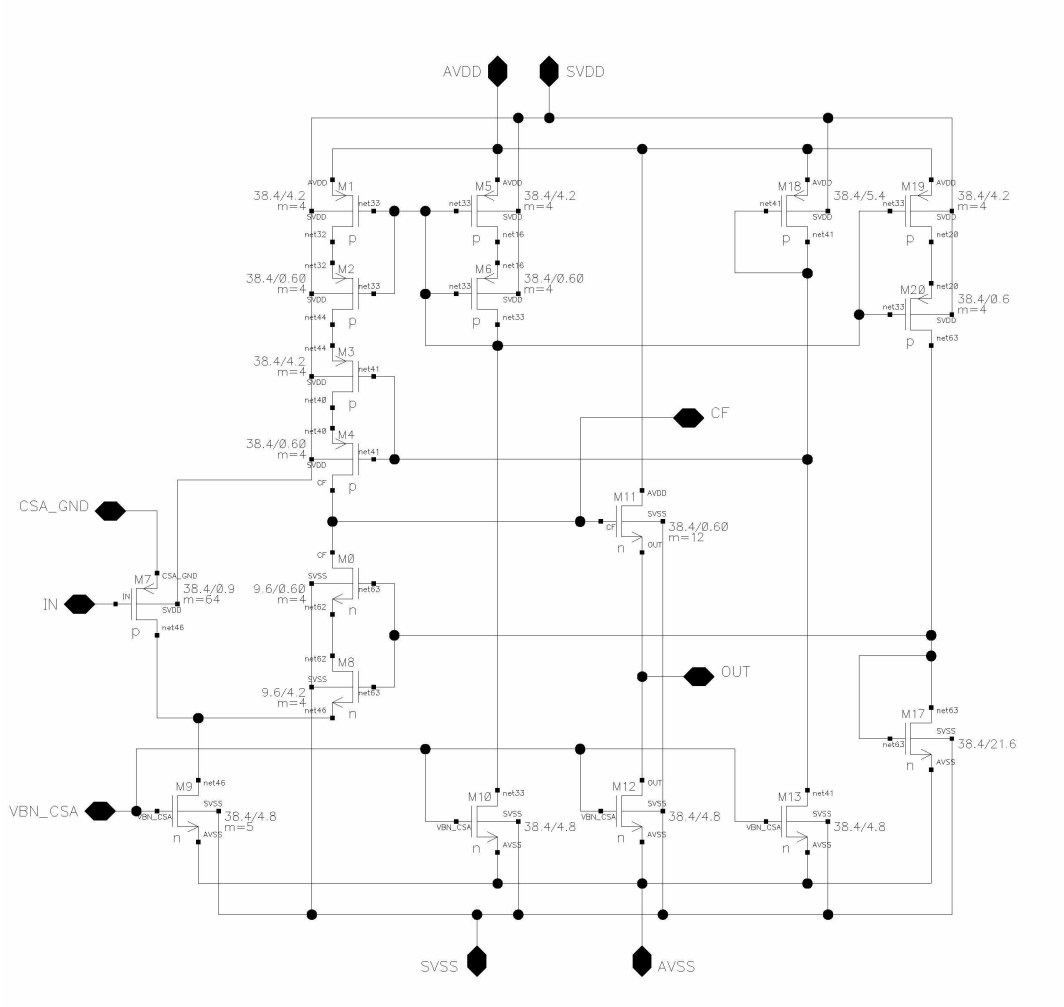


Figure 2.3 Schematic of the OTA used in the CSA.

The next step is to determine the value for the feedback capacitor and the feedback resistor. The two parameters that have to be considered are the maximum charge packet expected and the decay time. The feedback capacitance can be found out using the following equation,

$$C_f = \frac{Q_{\max}}{V_{out-\max}} \tag{2.5}$$

where $Q_{\max}$ is the maximum charge packet expected and $V_{out\text{-}\max}$ is the maximum peak output voltage desired. From the value of C_f, R_f can be calculated using the following equation,

$$R_f = \frac{\tau_f}{2.2C_f} \tag{2.6}$$

A reasonable peak to peak voltage of the output of the CSA is ±1 V. Using equation (2.5), C_f is 2.4 pF in high gain mode. When a decay time of 50 µs is used then from equation (2.6), R_f is calculated to be 9.1 MΩ. Consequently a C_f of 2.5 pF and an R_f of 10 MΩ were chosen yielding a decay time of 55 µs. Similarly, for the low gain mode the C_f of 12.5 pF and an R_f of 2 MΩ were chosen.

The final step in the design of the preamplifier used in the CSA for the required rise time t_r depends on the *GBW* of the amplifier. For a given detector capacitance, the *GBW* should be as large as possible. The sizes of the other transistors in the preamplifier should be determined based on the gain and the bandwidth requirements. The output stage of the opamp can be designed with the knowledge of the maximum peak to peak voltage (±1V) required at the output of the CSA.

The opamp was simulated and it was found that the open loop gain was approximately 86 dB. The phase margin of the opamp was found to be 53º with 30 dB closed loop gain.

2.5 Pulse Shaper

The primary function of the pulse shaper is to optimize the signal-to noise ratio (SNR) of the detector readout system. The step signal from the CSA will be the input to the pulse shaper, which acts as an integrator and stores the charges. The resultant is a narrow pulse and is used for further processing.

The purpose of the shaper is to provide a voltage pulse whose height is proportional to the energy of the detected particle. Nuclear radiation will emit charged particles of

different energy levels. By counting and analyzing the amplitudes of a series of voltage pulses at the output of the filter, the energy spectrum of the radiation can be determined. A Gaussian pulse shaping method gives a signal to noise ratio closest to the maximum that is theoretically possible. Hence the desired step response of the shaper is a Gaussian shaped pulse, with a corresponding frequency response that has Gaussian shaped bandpass characteristic.

The pulse shaper as shown in Figure 2.4 is built using two linear transconductors and a voltage amplifier. The pulse shaper needs to have a constant peaking time, the time needed for the shaper output to reach its peak amplitude and a reasonable well defined Gaussian shaped pulse over the expected range of inputs.

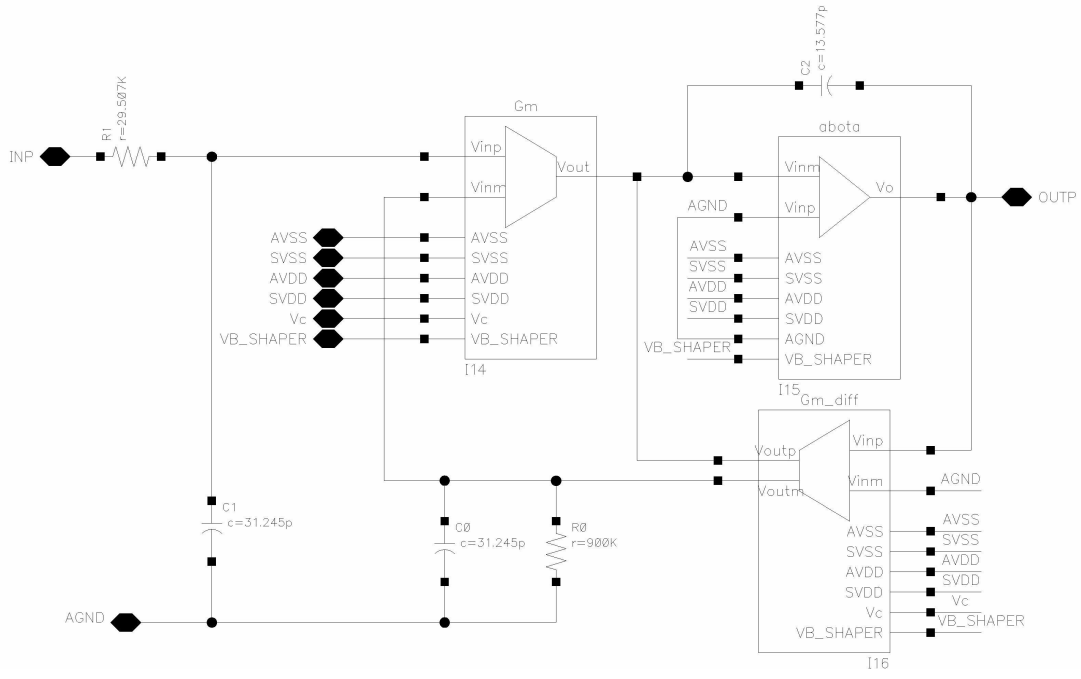


Figure 2.4 Block diagram of a pulse shaper.

The transconductors have high output impedance and are used as voltage controlled current sources. These transconductors produces an output current that is proportional to the

difference between the input voltages. As we use large signal transconductors, the difference between the inputs can be hundreds of millivolts and still be linear. The output amplifier has low output impedance and can drive resistive loads. It is used as an integrator.

2.5.1 Single ended linear transconductor

The OTA to be presented in this section, shown in Figure 2.5 is derived from the transconductor proposed in [3]. The transistors in this circuit operate in saturation and are characterized by the simple first order model:

$$I_D = \frac{\beta(V_{gs} - V_T)}{2.} \tag{2.7}$$

$$\text{where } \beta = \frac{\mu C_{ox} W}{L}. \tag{2.8}$$

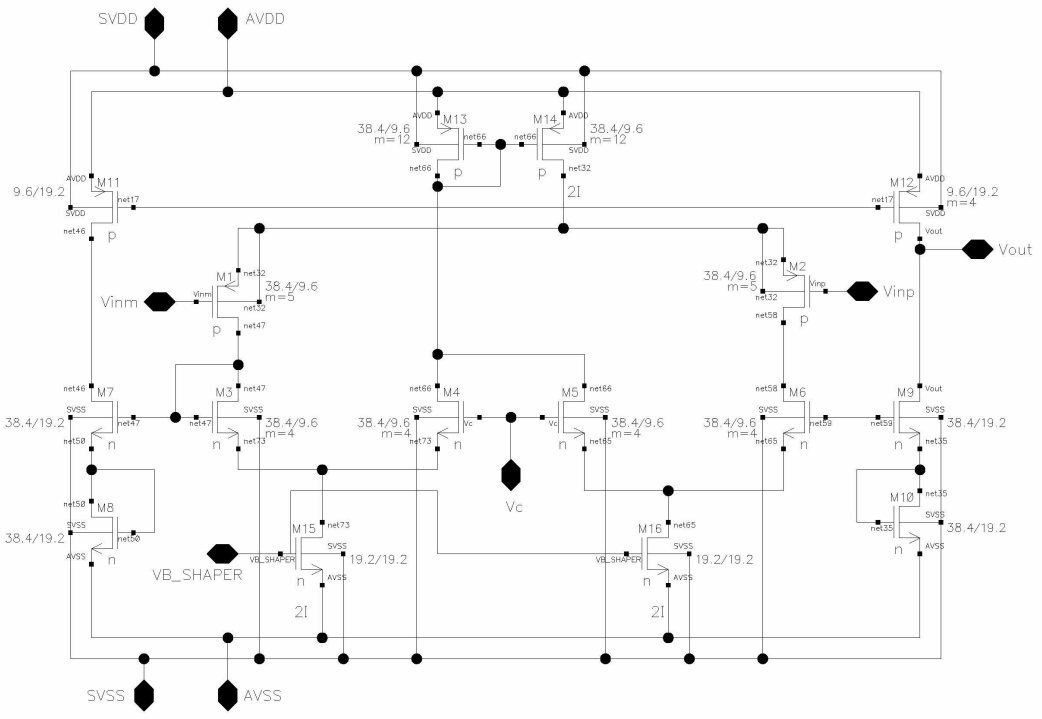


Figure 2.5 Schematic of the single ended linear transconductor used in the shaper.

As explained in [3], the differential pairs M_1/M_2, M_3/M_4, M_5/M_6 operate as attenuating current copiers, copying the input voltage V_{in} to the gate-to-source voltage of M_7 and M_9 with opposite signs, but with the same offset V_c . This scaled copy of V_{in} is converted to a differential current by the devices M_7 and M_9.

By means of the current mirror M_{11}/M_{12} this differential current is transferred to the output. The transconductance of the OTA as derived in [2] is

$$G_m = \frac{I_{out}}{Vin} = 2\beta_{n7,eq}\sqrt{\frac{\beta_{p1}}{\beta_{n3}}}(V_c - V_{tn,eq}) \qquad (2.9)$$

From equation (2.9) we can conclude that the G_m of the OTA can be varied by varying the control voltage (V_c). The $V_{gs,7}$ and $V_{gs,9}$ will be high because the value of V_c should be kept high for the current sources M_{15} and M_{16} to operate correctly. This is undesirable for the dc gain and the power consumption of the OTA. To decrease the effective $V_{gs,7}$ and $V_{gs,9}$, transistors M_8 and M_{10} were added in series with them.

From the analysis done in [3], one can come to a conclusion that stacked pair of identical transistor exhibit a better square law behavior than a single transistor.

2.5.2 Double ended linear transconductor

The double ended linear transconductor described in this section is similar to the single ended transconductor with an addition of two more legs. The schematic of the double ended linear transconductor is shown in Figure 2.6.

The current flowing through the output node V_{outp} is

$$I_{outp} = (I_1 - I_2)$$

Where I_1 is the drain current (I_d) of M_6 and I_2 is the drain current of M_{14}.

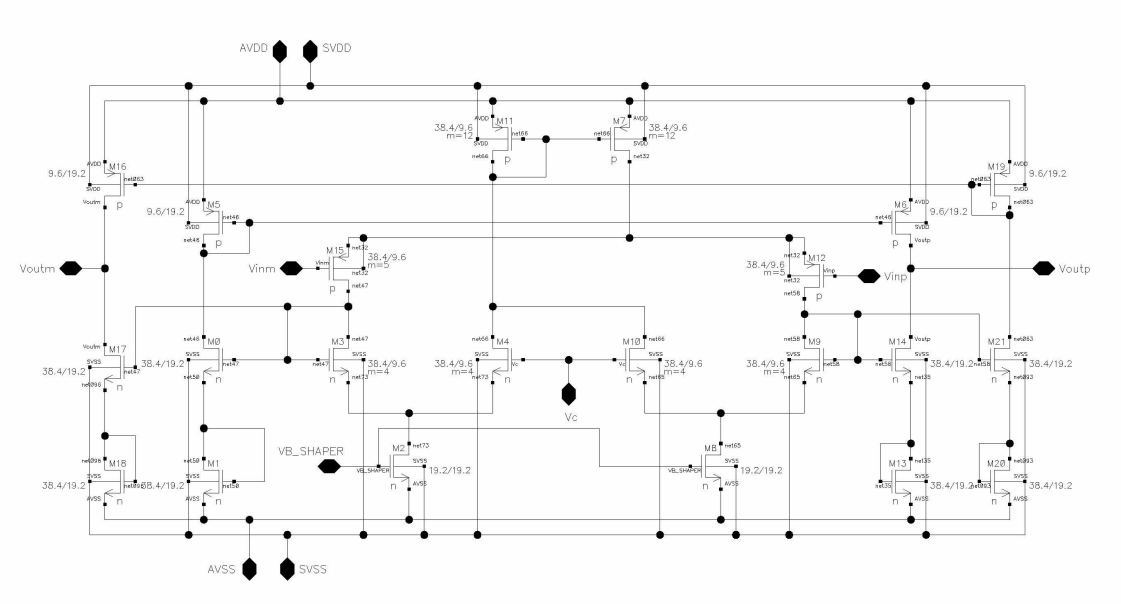


Figure 2.6 Schematic of the double ended linear transconductor used in the shaper.

The I_d of M_{16} is equal to I_2 due to the fact that M_{14}/M_{21} and M_{16}/M_{19} are current mirrors. Also, the I_d of M_{17} is equal to I_1 due to the current mirrors M_0/M_{17} and M_5/M_6 are current mirrors. Hence the current flowing through the output node V_{outn} is

$$I_{outn} = (I_2 - I_1)$$

Thus we have constructed a double ended linear transconductor.

2.5.3 Voltage amplifier

The amplifier used in the output stage of the pulse shaper operates as an integrator and is able to drive resistive loads. It is a voltage amplifier with a low impedance output stage and is shown in Figure 2.7. This amplifier consists of two gain stages and a class *AB* output stage.

Transistors M_{11} and M_{12} form the differential input pair. In order to reduce the flicker noise, these transistors are designed with large areas. The current mirror M_{19}/M_{20} biases the

transistors M_{11} and M_{12}. The sizes of the other transistors in the input stage are determined by the gain requirement. To obtain a relatively large phase margin, the so called grounded gate cascade compensation is used in this stage. In this method, the phase compensation capacitor C_0 between the input and the output stages is disconnected to the input stage and is connected to a virtual ground node (*AGND*) by the source node of a common gate cascaded transistor M_7. As a result, the feed forward path that causes the performance degradation at high frequency is removed from the circuit. The dominant pole related to the phase compensation capacitor can still be created by feeding the first stage a feedback current that is independent of the first stage output. Transistors M_{15} and M_6 are used to provide the second stage with a relatively large input dynamic range as they are common-source connected for the signal.

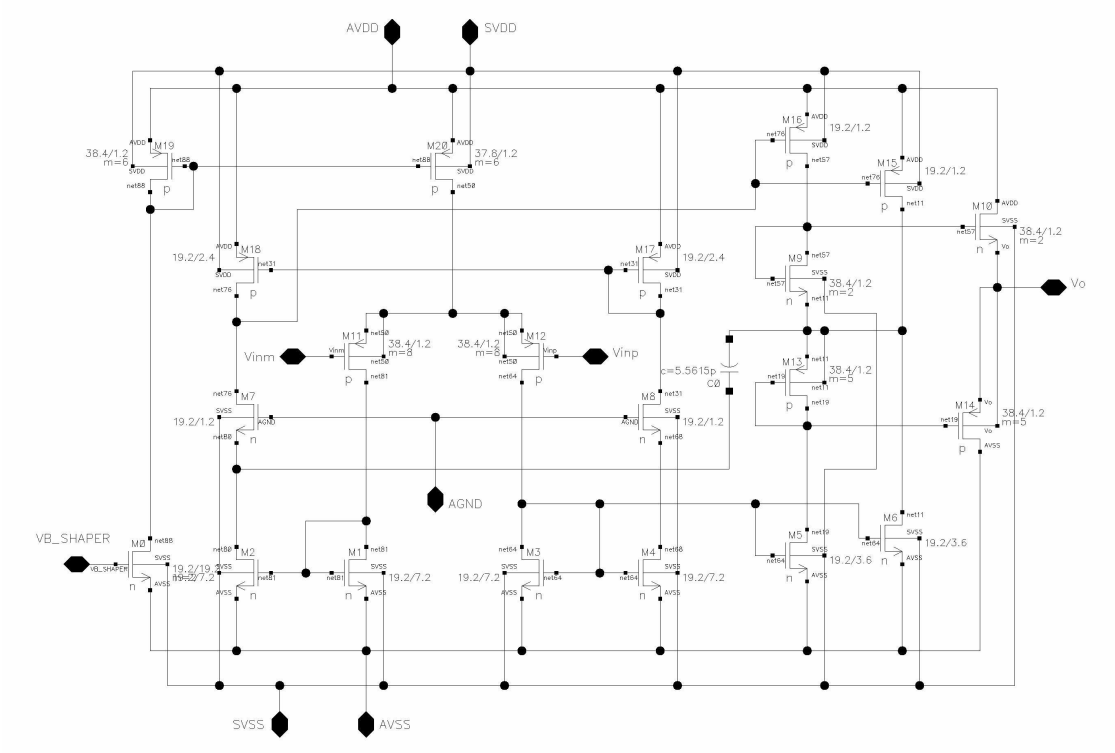


Figure 2.7 Schematic of the voltage amplifier used in the shaper.

The phase compensation capacitor C_0 is connected to the drain of M_{15} (M_6), which is also the input node of the second stage.

The gain of the first stage is given by the product of $g_{m,\,M11}$ and $r_{ds,\,M19}$. Analytical results show that the gain of the second stage is given by, $-\frac{g_{m,M16}+g_{m,M15}}{g_{ds,M10}+g_{ds,M14}}$. The high impedance point at the drain of M_{18} determines the corner frequency of the amplifier. The capacitance at this node is the sum of the diffusion capacitance and the compensation capacitance C_0. Due to the miller effect the capacitance at this node appears to be a large capacitance to ground. This helps in reducing the dominant pole frequency and hence increases the stability of the amplifier.

2.6 Peak Sampler

The peak sampler shown in Figure 2.8 amplifies the signal from the pulse shaper and detects the peak of the amplified signal for both the positive and the negative pulses. The output of the peak sampler is a measure of the energy of the input particle. The peak sampler

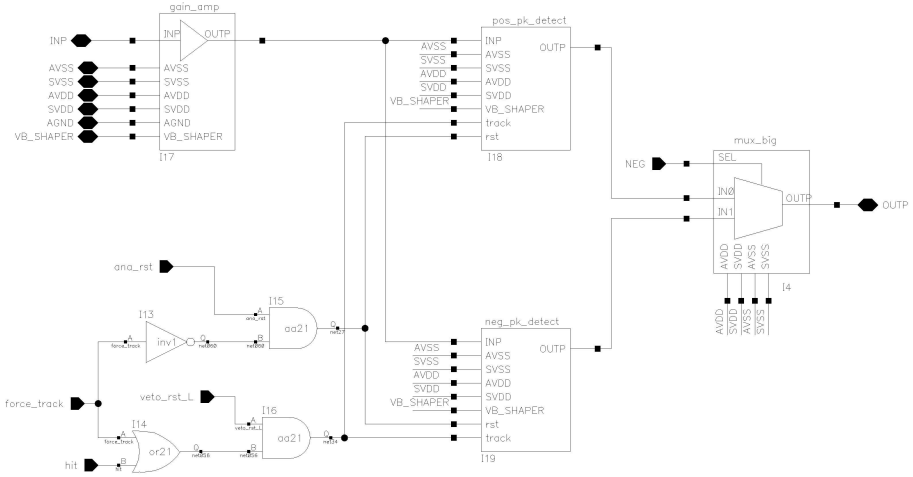


Figure 2.8 Block diagram of peak sampler.

consists of the gain amplifier, positive peak detector, negative peak detector and a multiplexer. The multiplexer selects between the outputs from the positive or the negative peak detector depending on the input to the system.

2.6.1 Non-linear gain amplifier

The non-linear gain amplifier used in the peak sampler circuit is a compact opamp with miller compensation. The opamp shown in Figure 2.9 consists of a rail-to-rail input stage and a rail-to-rail class AB output stage. The opamp is compensated using the conventional miller technique. The capacitor CM1 and CM2 around the output transistors, M8 and M31, split apart the poles ensuring a 20 dB per decade roll off of the amplitude characteristic.

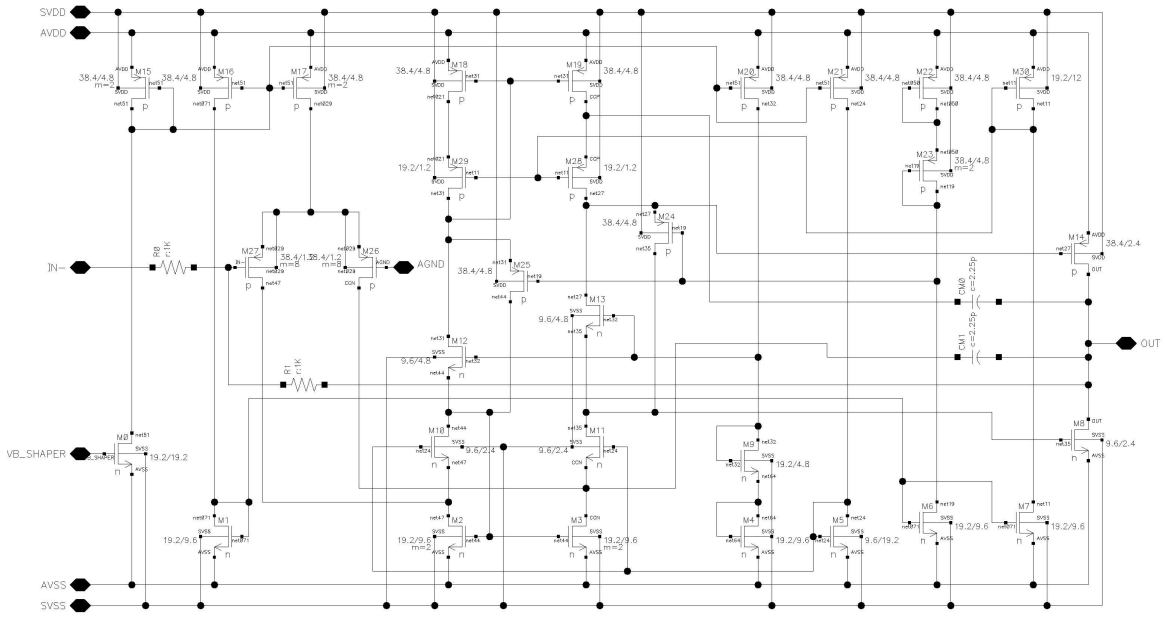


Figure 2.9 Schematic of non-linear gain amplifier

The conventional miller shifts the output pole up to a frequency of approximately

$$\omega_{out} = \frac{g_{m0}}{C_L} \frac{C_M}{C_{gs,out}} \tag{2.10}$$

where g_{m0} is the transconductance of the output transistor, C_L is the load capacitor, C_M is the total miller capacitor and $C_{gs,\ out}$ is the total gate-to-source capacitance of the output transistor. The gain of the opamp is determined by the resistors R_1 and R_2. In this case the gain of the opamp is 4 (12 dB).

2.6.2 Positive peak detector

The positive peak detector gets its input from the gain amplifier discussed in the previous section. The positive detector shown in Figure 2.10 tracks the peak of positive pulses at its input. Current mirrors *M12/M13* and *M10/M11* ensure the proper biasing for the circuit. The output tracks the peak only when the *track* signal is low and vice versa.

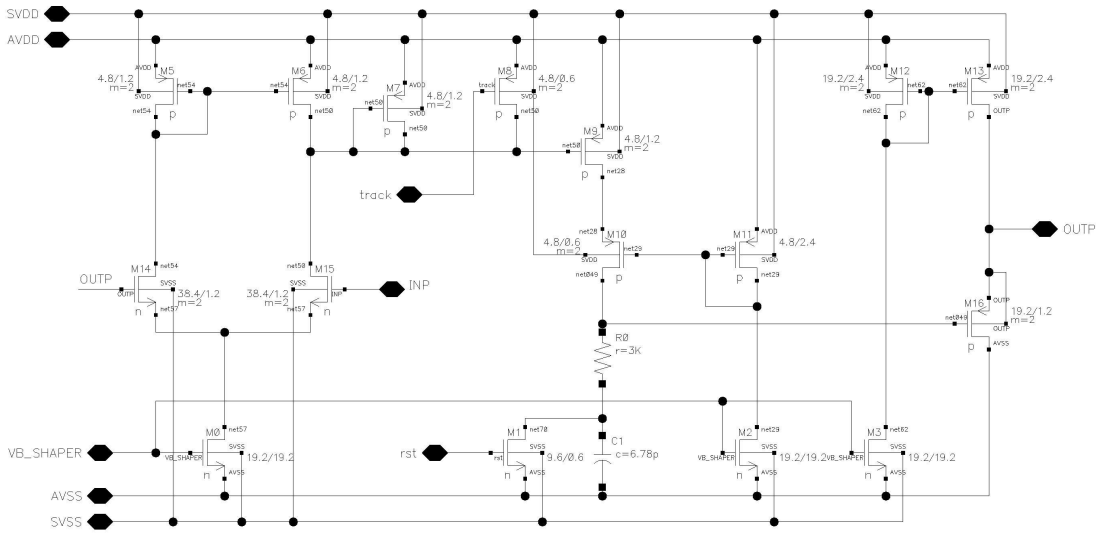


Figure 2.10 Schematic of positive peak detector circuit

In track mode the capacitor C_0 is charged, which is proportional to the amplitude of the input signal *INP*. The output voltage is proportional to the charge in the capacitor. This output is

fed back to one of the differential pair inputs. Once the peak voltage is reached the input starts decreasing which causes the output to hold the peak amplitude.

2.6.3 Negative peak detector

The negative peak detector works similar to the positive peak detector discussed in the above section except for it holds the peak of a negative going pulse only.

2.7 Nowlin Circuit

The Nowlin circuit is shown in Figure 2.11. The capacitor C_1 at the input is used to block DC voltage. The circuit which converts the unipolar signal to a bipolar signal is an all pass filter that is built using a low pass filter (R_1, C_2) and a voltage divider (R_2, R_3+R_4). If a V_{in} is the input to the filter, then V_{outp} can be expressed as

$$V_{outp} = \left(\frac{\omega_0}{s + \omega_0} - k \right) \times V_{in} \tag{2.10}$$

where ω_0 is the cut-off frequency of the low pass filter and k is the voltage divider ratio. If we assume k = ½ then we get an all pass filter.

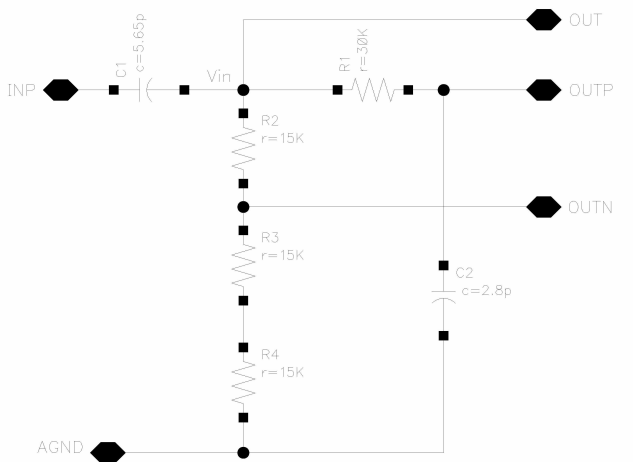


Figure 2.11 Schematic of the nowlin circuit.

The input to all pass filter, V_{in} can be approximated to a step input. The output of the nowlin circuit is shown in Figure 2.12.

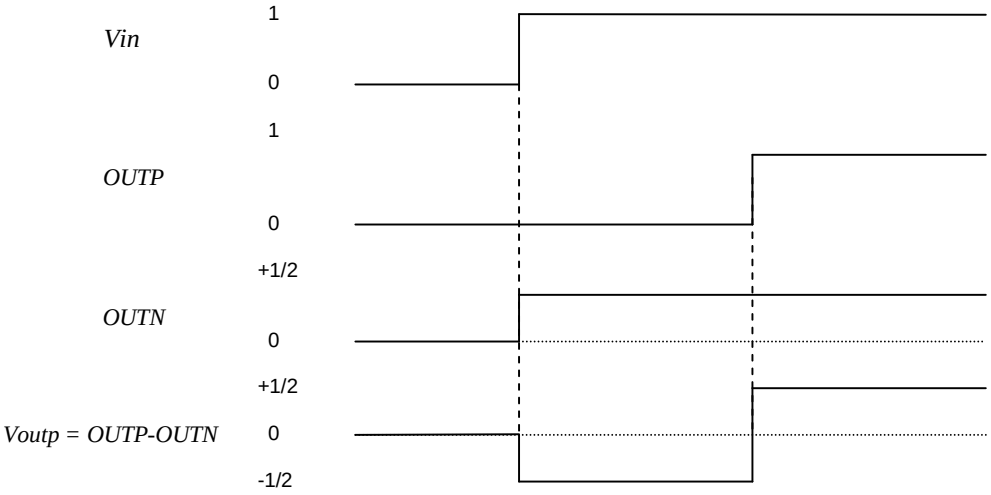


Figure 2.12 Signals at various nodes of the nowlin circuit.

2.8 Pseudo Constant Fraction Discriminator

To develop a logic signal that is indicative of the arrival time of the particle at the detector or the output of the CSA, a discriminator is used. A simple leading edge discriminator alone cannot be used in this application since a signal with 80 ns risetime would generate a walk of tens of nanoseconds. The error in the timing should not be more than ±0.5 ns. So a constant fraction discriminator (CFD) is used. As shown in Figure 2.13, the CFD contains two discriminators that are used to perform the timing measurement. One is the zero crossing discriminator that fires on the zero crossing of the bipolar signal and generates the precise timing signal. Since this discriminator is set to fire as soon as the input signal crosses zero, it is susceptible to noise firings. The function of the second discriminator – commonly known as the leading edge discriminator – is to block these noise firings and

only allow an output when a valid signal is present. The final output of the CFD is a positive-going pulse with a fixed pulse-width generated by a one-shot circuit.

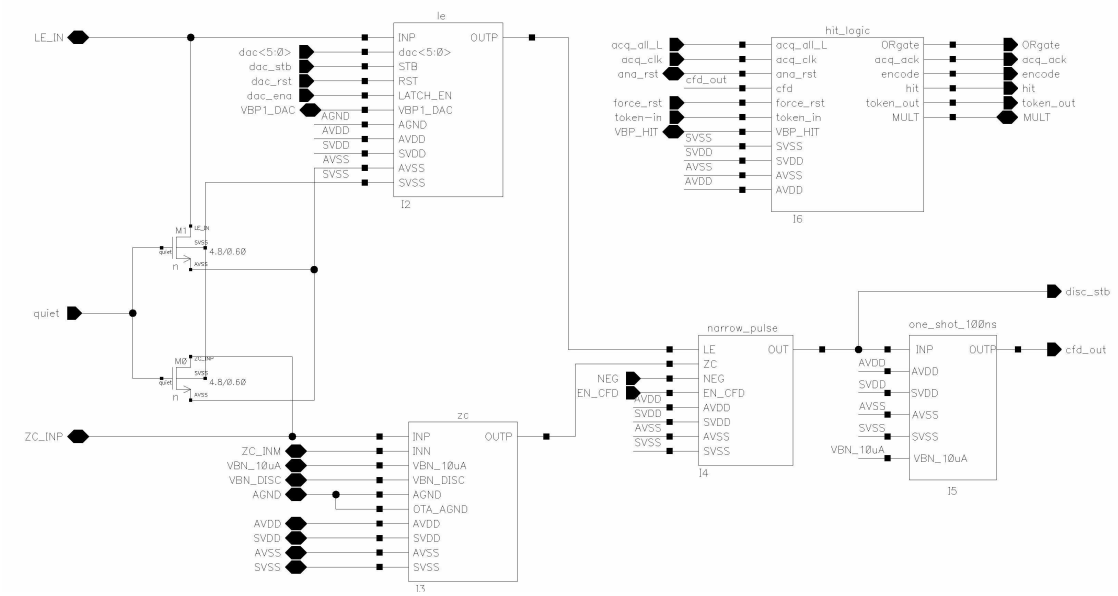


Figure 2.13 Block diagram of the CFD.

2.8.1 Leading edge discriminator

As shown in Figure 2.14, the leading edge discriminator is composed of cascaded stages of differential input/differential output amplifiers, a digital-to-analog converter (DAC) and in the final stage a comparator. The DAC serves to nullify the offset voltage present at the input of the differential amplifier and also to set a threshold voltage so that the CFD does not fire.

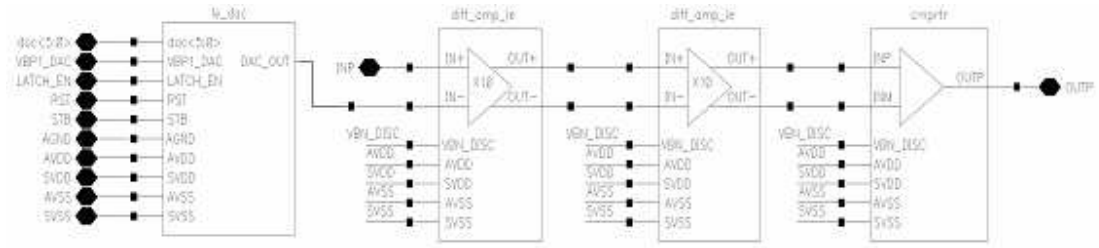


Figure 2.14 Block diagram of the leading edge discriminator

2.8.1.1 Digital-to-analog converter (DAC)

The digital-to-analog converter (DAC) is used to correct offsets associated with the leading edge discriminator is shown in Figure 2.15. It consists of a binary weighted array (weights: 1, 2, 4, 8, 16) of current sources and a binary weighted array (weights: 1, 2, 4, 8) of current sinks. The most significant bit, bit 5, indicates the algebraic sign and whether the current sources or current sinks are used. In other words, the data format is sign/magnitude with 5 bits of magnitude in the positive direction and 4 bits in the negative direction.

An output voltage is created by either sourcing or sinking current through a 0.5 kΩ resistor (R_0 or R_1), connected to analog ground (AGND = 2.5 VDC). The required biasing voltage is generated from the voltage source VBP1_DAC. A maximum positive

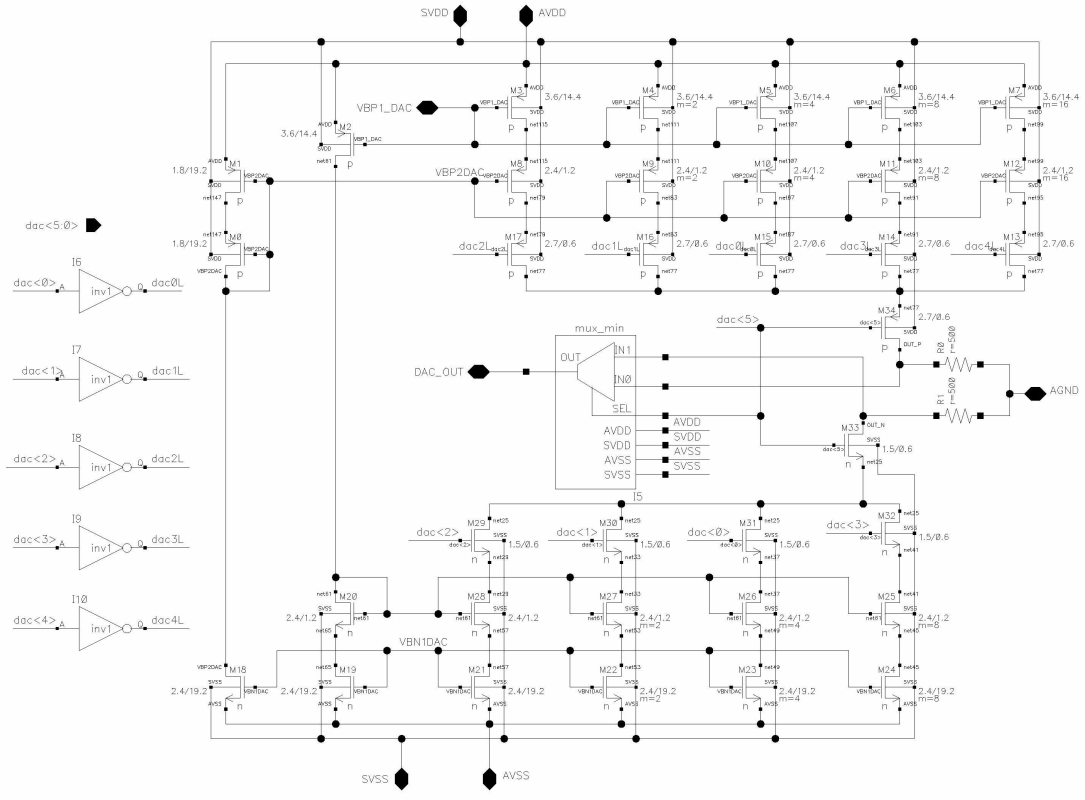


Figure 2.15 Schematic of the DAC used in the leading edge discriminator.

output voltage of 19.375 mV with respect to *AGND* can be achieved. The most negative output is -9.375 mV (relative to *AGND*). The step size is approximately 0.605 mV. Settling time (better than 1 %) on the DAC outputs is 1 µsec. The step size of 0.605 mV corresponds to roughly 11,500 electrons. Offsets associated with the leading edge discriminator are expected to be 10 mV (3 sigma). This implies that we are guaranteed to be able to set maximum thresholds at least at the 178,000 electron level.

2.8.1.2 Differential amplifier

The differential amplifier as shown in Figure 2.16 is composed of a differential input pair (M_0, M_1) loaded with diode connected transistors (M_9, M_{10}). Proper biasing to the circuit is provided by the current mirror M_6/M_7. The gain of this stage is given by the ratio of the transconductance, g_m of the transistors in the differential pair and the g_m of the diode connected transistors. The g_m of M_0 is 500 µmhos and that of M_6 is 42 µmhos, yielding a gain of approximately 10 (20 dB). The bandwidth is about 110 MHz and the input offset is 5 mV.

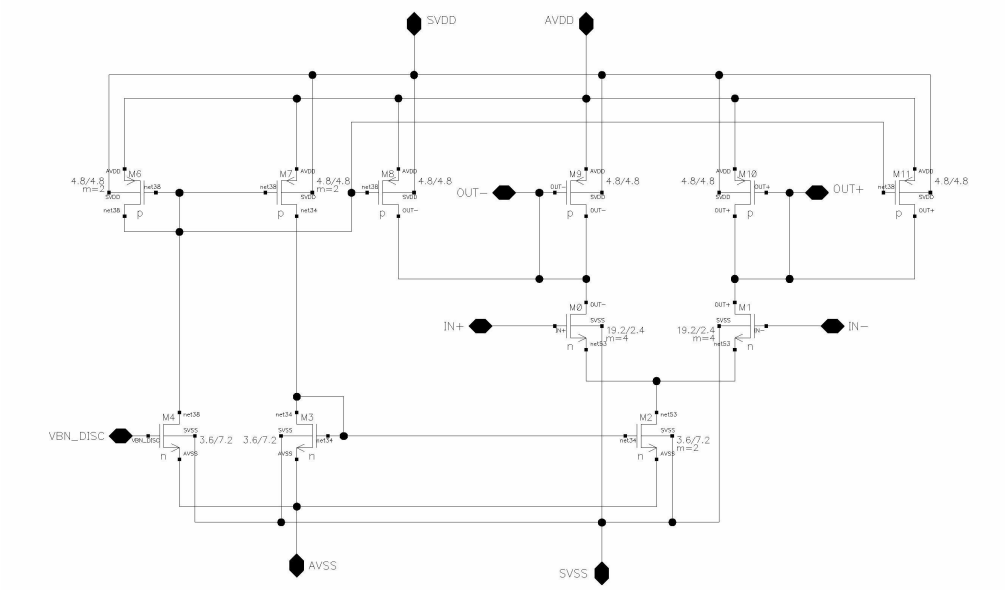


Figure 2.16 Schematic of the differential amplifier used in Leading edge discriminator.

2.8.1.3 Comparator

The design of the comparator used in the leading edge discriminator is shown in Figure 2.17. The output of the comparator is pulled to the positive rail of the supply voltage if the voltage at *INP* is greater than *INM* and vice versa. The current mirror M_{17}/M_{18} provides the bias currents needed for the circuit. Let's assume that the voltage at *INM* is greater than the voltage at *INP*. This tends to lower the voltage at node *X* and raise the voltage at node *Y*. This in turn pulls the voltage at node *X* towards the positive supply rail due the positive feedback present in the circuit. A large enough differential signals at the input will drive the output to the high state or a low state depending on the polarity of the differential input. The input offset for this circuit is on the order of 10 mV.

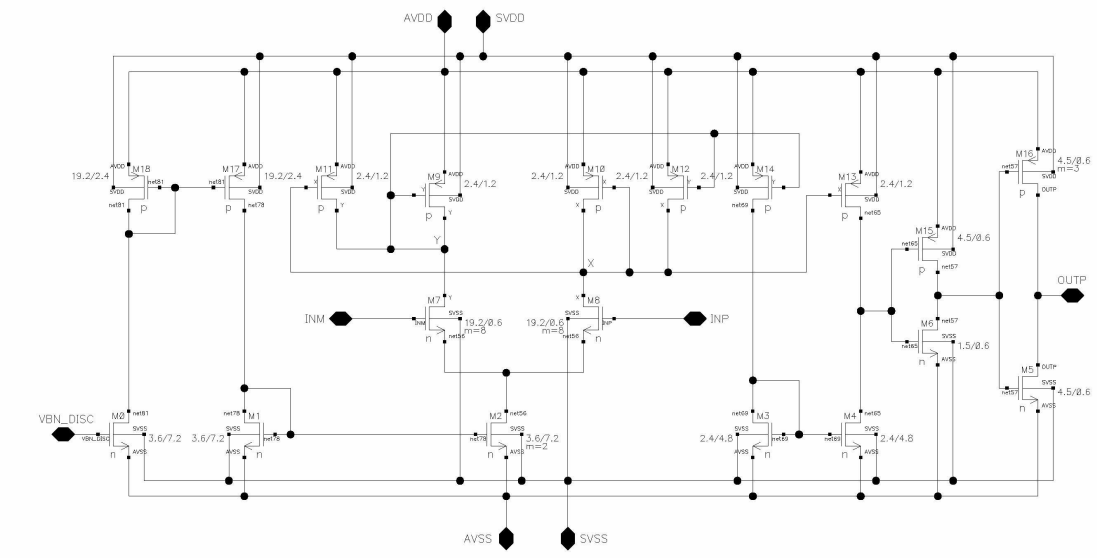


Figure 2.17 Schematic of the comparator used in Leading edge discriminator.

2.8.2 Zero crossing discriminator

Figure 2.18 is a detailed block diagram of the zero crossing discriminator. It is composed of cascaded stages of differential input/differential output amplifiers. The gain-

bandwidth product of this circuit increases with each additional stage in the cascade. This arrangement does not necessarily provide the lowest propagation delay through the circuit, but it does minimize the spread in propagation delay as a function of the input amplitude. The final stage of the circuit is the comparator which produces a CMOS-level logic signal.

Also included in the zero crossing discriminator is a continuous feedback for removing the dc offset of the comparator.

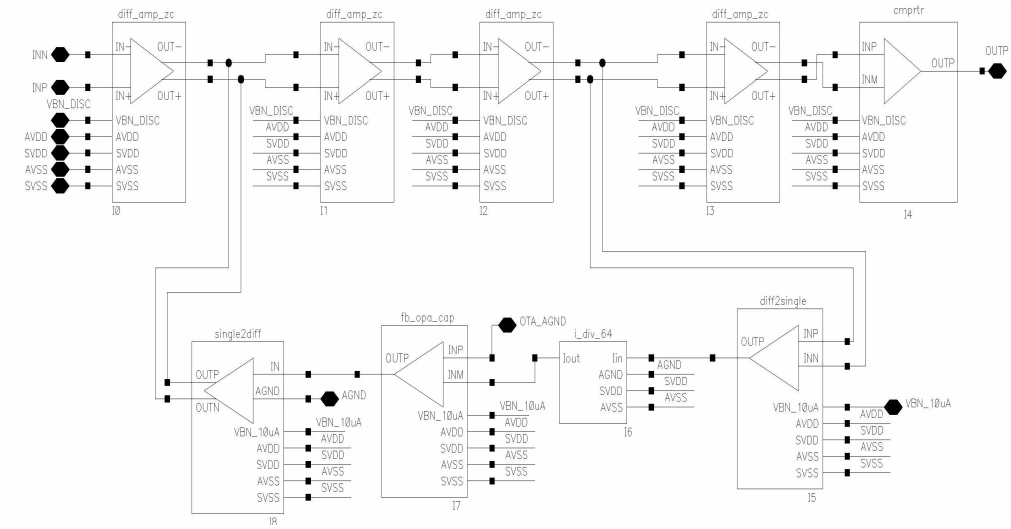


Figure 2.18 Block diagram of the zero crossing discriminator

2.8.2.1 Differential amplifier

The differential amplifier used in the zero crossing discriminator is similar to the one used in the leading edge discriminator. The only difference is in the g_m of the diode connected transistor and the input differential transistors. This changes the gain of the amplifier to approximately 7.5 (17.5 dB). The bandwidth of the amplifier is 100MHz and the input offset is in the order of 17 mV.

2.8.2.2 Comparator

The comparator used in the zero crossing discriminator is same as in Figure 2.17 that

is used in the leading edge discriminator.

2.8.2.3 DC offset cancellation

The DC offset cancellation block is the feedback loop that can be seen in Figure 2.18. The diff2single block is a transconductance amplifier, which converts the differential input voltage into current. The i_div_64 is just a current attenuator block that divides the input current by a factor of 64. The combination of the diff2single and the i_div_64 blocks is a better way for building a large resistor. The effective resistance is calculated using R_{eff} = 64/$g_{m,}$ where g_m is the transconductance of input device in diff2single block. The value of effective resistance is found to be approximately 2 MΩ.

The resistor along with the operational amplifier (fb_opa_cap) will form an integrator, whose corner frequency is at 20 KHz. The single2diff block is used to null out the offset present in the first stage of the zero crossing discriminator.

2.8.3 Narrow pulse

The narrow pulse circuit shown in Figure 2.19 is used to convert the output of the zero crossing discriminator and the leading edge discriminator into a narrow digital pulse to

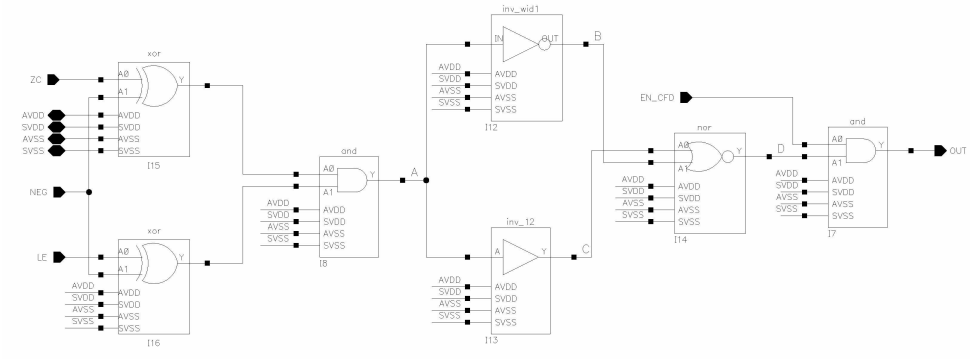


Figure 2.19 Block diagram of the narrow pulse circuit used in CFD.

be processed by the one shot in the CFD.

The inputs to the narrow pulse circuit can be approximated to a digital step input. The intermediate signals at various nodes in the circuit are shown in Figure 2.20. The signal *B* is the inverted version of signal *A* and signal *C* is the delayed version of signal *A*, obtained by passing signal *A* through a series of inverters. The signal *C* and signal *B* are passed through a *NOR* gate to obtain the final narrow pulse.

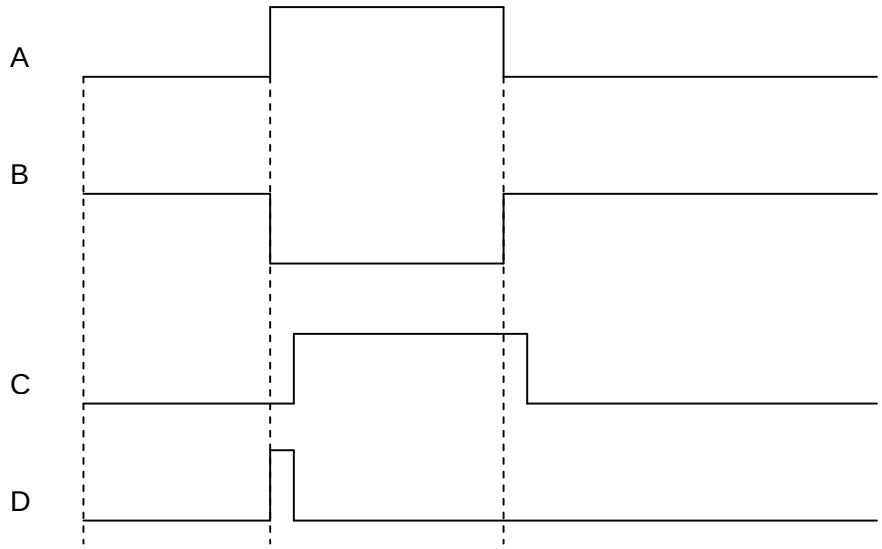


Figure 2.20 Signals at various nodes in the narrow pulse circuit.

2.8.4 One shot

The one shot circuit used in the CFD is shown in Figure 2.21. The one shot converts the output from the narrow pulse circuit to a pulse with a larger pulse width. The main components of the circuit are the Schmitt trigger and the latch. The input pulse sets the latch which is the output to the circuit. The output going high will cause the capacitor C_0 to discharge and thus the input to the Schmitt trigger also lowered. The Schmitt trigger is designed such that it fires when the input goes below 1.5V. The firing of the Schmitt trigger

will reset the latch causing the output of the one shot to go low. The width of the output pulse is calculated as follows.

$$T = \frac{C \times \Delta V}{I} \tag{2.10}$$

where C is the value of the capacitor C_0, ΔV is the change in the voltage before the Schmitt trigger fires and I is the current through the biasing circuit. The value of C_0 is

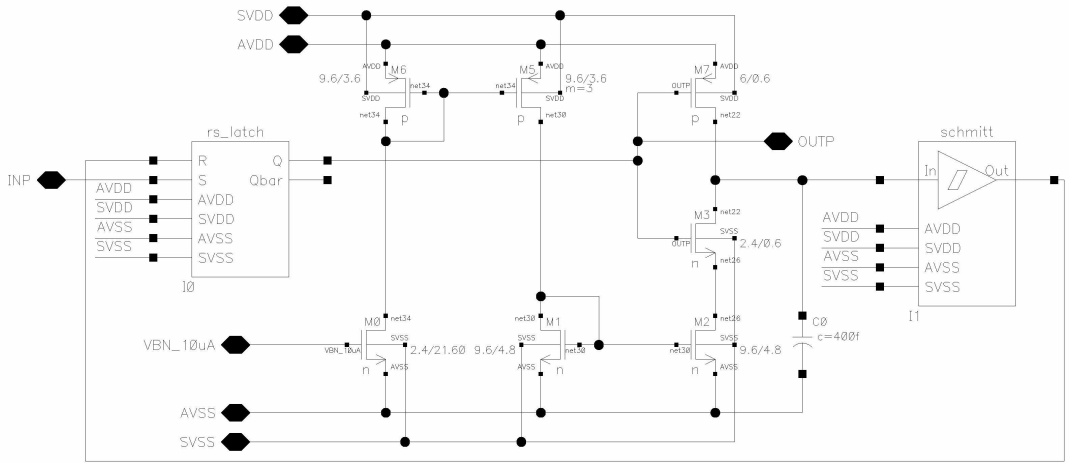


Figure 2.21 Block diagram of the one shot.

400 fF, ΔV is 3.5V and the current produced by the current mirror M_1/M_2 is 15uA which produces an output with a pulse width of approximately 80 ns.

2.8.5 Hit Logic

Figure 2.22 shows the block diagram of the hit logic circuit present in the CFD. It consists of two registers: the hit register and the active register. The hit register gets set when the CFD fires, indicating that the circuit is hit. This register is reset only when all the useful data from that particular channel is readout or it can be forced to reset by an external reset signal. The data is readout from the channel only when a token is passed in to the circuit and

when the data is read this token is passed in to the next channel. This token is passed sequentially till it reaches the final channel in the chip. This circuit consists of a pulldown transistor whose drain node is connected to the same node in all channels and to a pullup trtansistor, thus creating a pseudo type OR gate. The output from this node is an indication of at least one channel in the chip being hit. It also houses a part of the total multiplicity circuit, from which one can determine the number of channels that are hit.

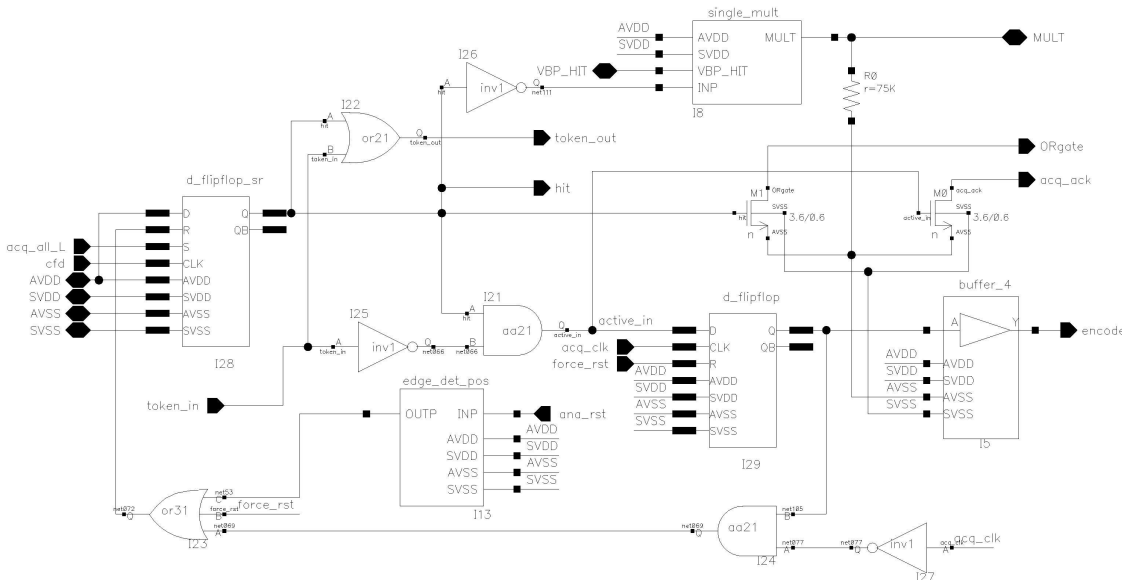


Figure 2.22 Block diagram of the hit logic circuit.

2.9 Time-to-Voltage Converter

The timing details of a particular event of interest in the channel can be measured using the time-to-voltage converter (TVC) shown in Figure 2.23. The TVC is designed to operate in two different mode settings, the 250ns/V mode and the 1µs/V mode. The TVC can be switched between these two modes by changing the bias voltage *VB_TVC*. The measurement is stored as an analog voltage on a capacitor. The proper biasing for the circuit is provided by the current mirrors M_0/M_2 and M_5/M_8. When the channel is not being hit the

transistor M_7 turned on and the transistor M_6 is turned off. During this condition all the current flows through M_7 and no current through M_6. When the channel is hit, transistor M_6 is turned on and M_7 is turned off which diverts all the current through M_6, thus charging the capacitor C_0 . The output stage of the TVC is an n-type source follower which reflects the

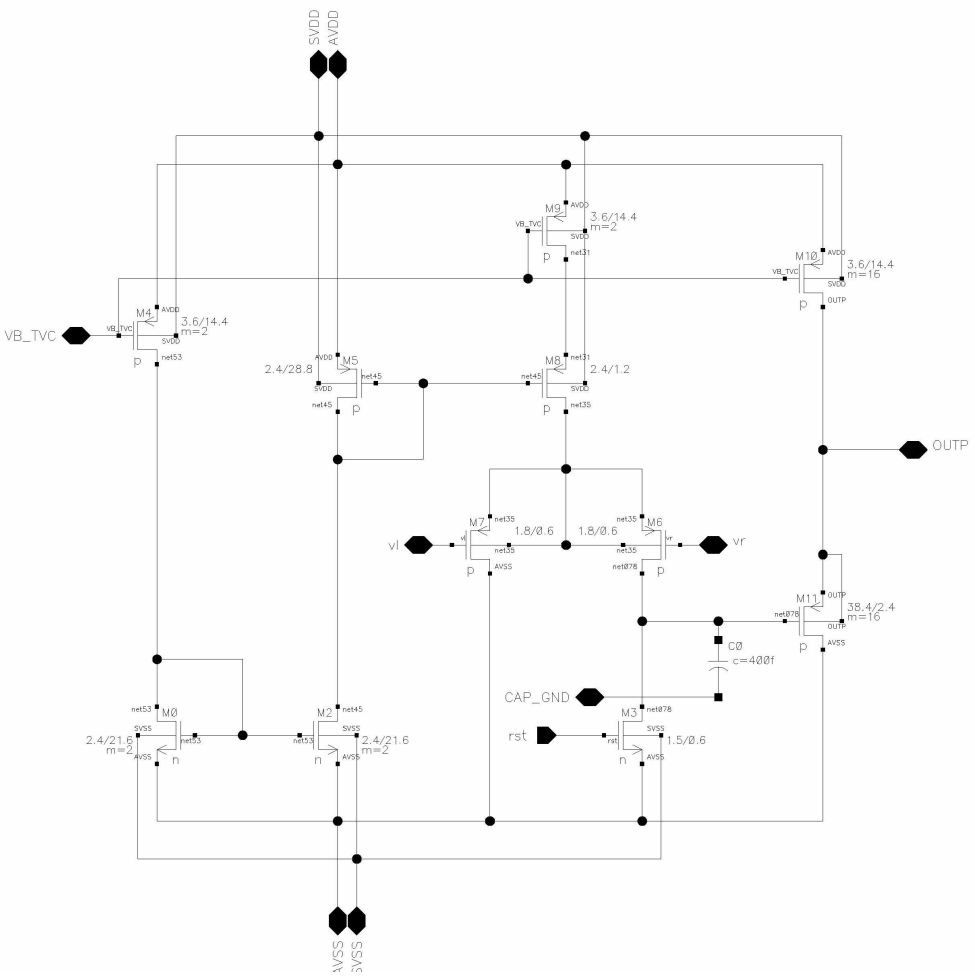


Figure 2.23 Schematic of the time-to-voltage converter circuit.

voltage in the capacitor C_0.The amount of charge in the capacitor gives the time of hit in the channel. Discharging the charge in the capacitor will take place through the transistor M_3.

When the channel is hit, the current flowing through M_7 should be steered slowly into M_6 as this will reduce a large change in the node voltage. This can be done only if both the

transistors are turned on for some period of time. This function is taken care of by the circuit shown in Figure 2.24 and is called the width generator. The circuit is built with a few logic gates and a couple of latches. This circuit generates input signals for M_6 and M_7 such that both the transistors are turned on for a short period of time. The circuit inputs start and stop signals mark the beginning and the end of the time interval to be measured. The default output signals from the circuit are *vl* being low and *vr* being high .When there is a hit in the channel, the start signal goes high and this brings *vr* low. This signal is passed through an inverter which delays the *vr* signal and is used to reset the other latch which brings *vl* high. The time during which both *vr* and *vl* are low depends on the delay through the inverter. The size of the inverter is selected accordingly to produce an overlap time of atleast 2ns.

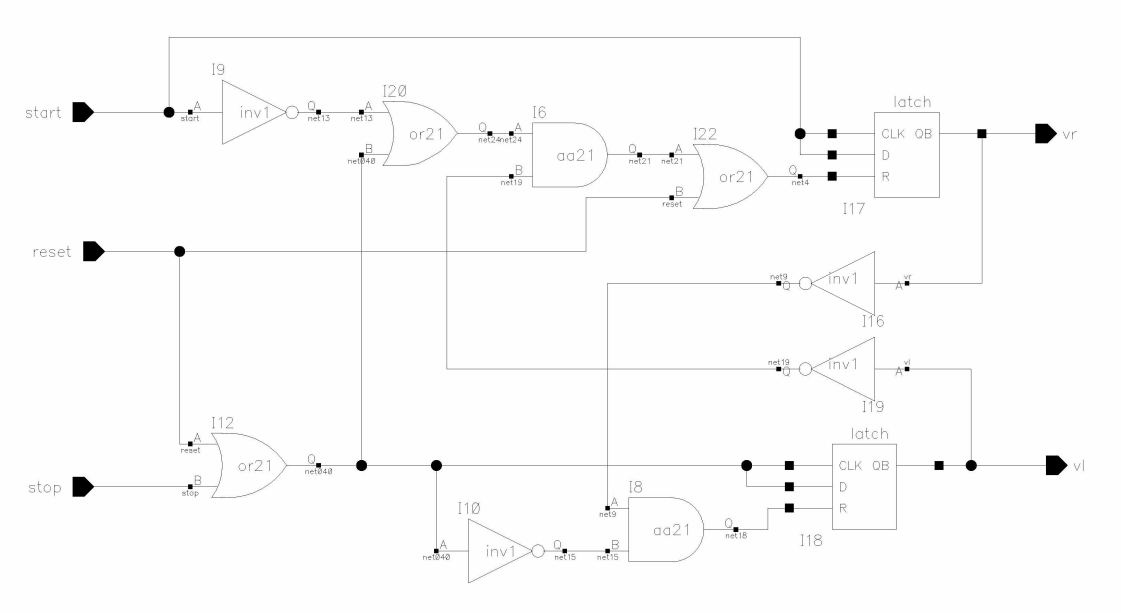


Figure 2.24 Schematic of the width generator.

At the beginning of the measurement the following chain of events takes place in quick succession: *a*) initially, M_7 is on, M_6 is off and no current flows to the capacitor; *b*)

then, after the start signal is received, M_6 is turned on before M_7 is turned off, and the two transistors supply current for a short time; *c*) finally, M_7 is turned off completely, and all of the current flows through M_6. This sequence of events is described in Figure 2.25. The resulting capacitor voltage will contain the components: a linear portion which is proportional to the time interval, and an offset which will depend on the length of time that both transistors are on. However, as long as a stable set of overlapping gate pulses are generated for M_6 and M_7 by the width generator, the offset will be a constant.

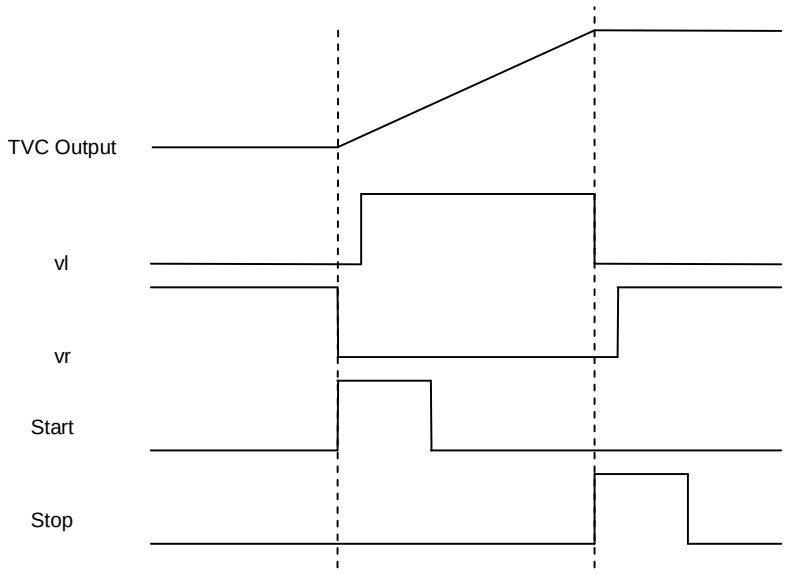


Figure 2.25 Sequence of events in the TVC.

2.10 Analog Reset Logic

Automatic reset circuitry has been incorporated into each channel and is shown in Figure 2.26. As discussed in section 2.8, when a channel is hit, the channel's CFD emits a positive-going pulse (generated by a one-shot circuit with a fixed pulse-width) that lasts for approximately 80 ns. This pulse is then used to trigger a second one-shot in the reset logic circuit. This second monostable circuit produces a negative-going pulse with a variable pulse

width. The delay, common to all channels on the IC, can be varied by controlling the voltage on the *DLY_VC* pin. The delay can be reliably varied from a few hundred nano-seconds to around 100 µsec by varying the control voltage.

If the *veto_rst* signal is not asserted prior to the trailing edge of the pulse from this second one-shot, the channel will reset itself. The following events will take place. The hit register in the CFD will be cleared, the time-to-voltage converter will be reset by discharging the internal capacitor, and the peak-sampling circuit will be reset. The *veto_rst* signal must remain asserted until the variable delay time has elapsed. It should also be noted that if the input pulse exceeds 100 ns then the output will produce multiple pulses.

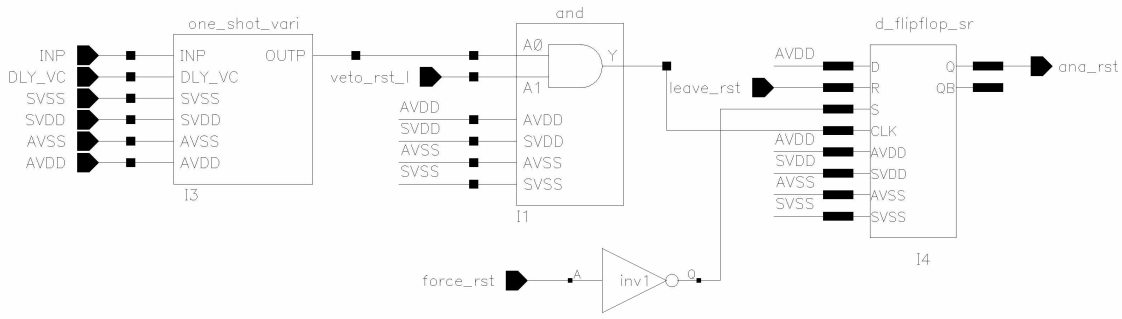


Figure 2.26 Block diagram of the reset logic.

2.11 Common Circuits

In addition to the 16 analog channels there exists a 17th channel in the IC. This additional channel contains the common bias circuits along with digital circuits common to all of the channels. This common digital circuitry and the bias circuits shown in Figure 2.27 will be described in this section.

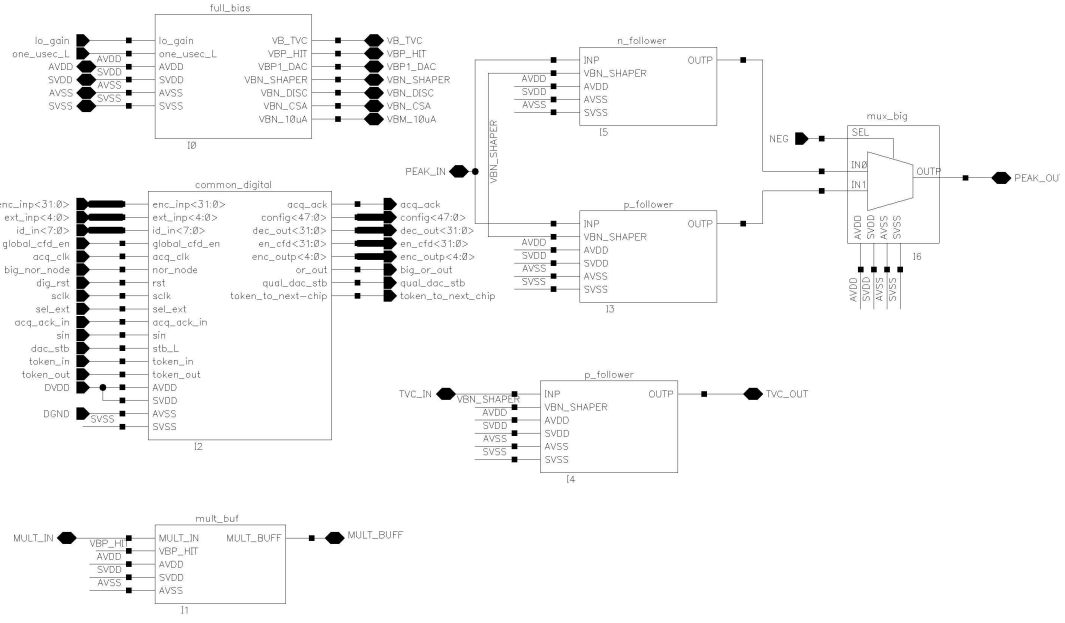


Figure 2.27 Block diagram of the common channel

2.11.1 Bias circuits

Several circuits are required to correctly bias the analog subsystems. This consists of a bandgap voltage reference, a constant current reference, and a reference for the digital-to-analog converter used to correct offsets associated with the leading edge discriminator.

2.11.1.1 Bandgap voltage reference

The core of the bias circuitry is a simple bandgap voltage reference. The bandgap reference shown in Figure 2.28 produces a relatively stable voltage (1.23 volts) with respect to both power supply and temperature variations.

The bandgap circuit makes use of parasitic bipolar vertical PNP transistors. A PTAT (Proportional to Absolute Temperature) current is created by forcing equal currents through two diode-connected transistors whose area differs by a factor of ten. A 60 mV voltage

exists across a 590 Ω resistor, producing a current of approximately 125 µA. The current is mirrored and passed through a 5.3 KΩ resistor to yield a voltage and summed with a base-emitter voltage. The base-emitter voltage displays a negative temperature coefficient and compensates the positive temperature dependence of the voltage developed across the 4.5 KΩ resistor.

Through the use of current mirrors, bias voltages corresponding to several different bias currents are also created. These bias voltages are then heavily filtered in order to greatly improve noise performance. The signal *lo_gain* is used to control the bias voltage required for the CSA depending on the mode it is operating.

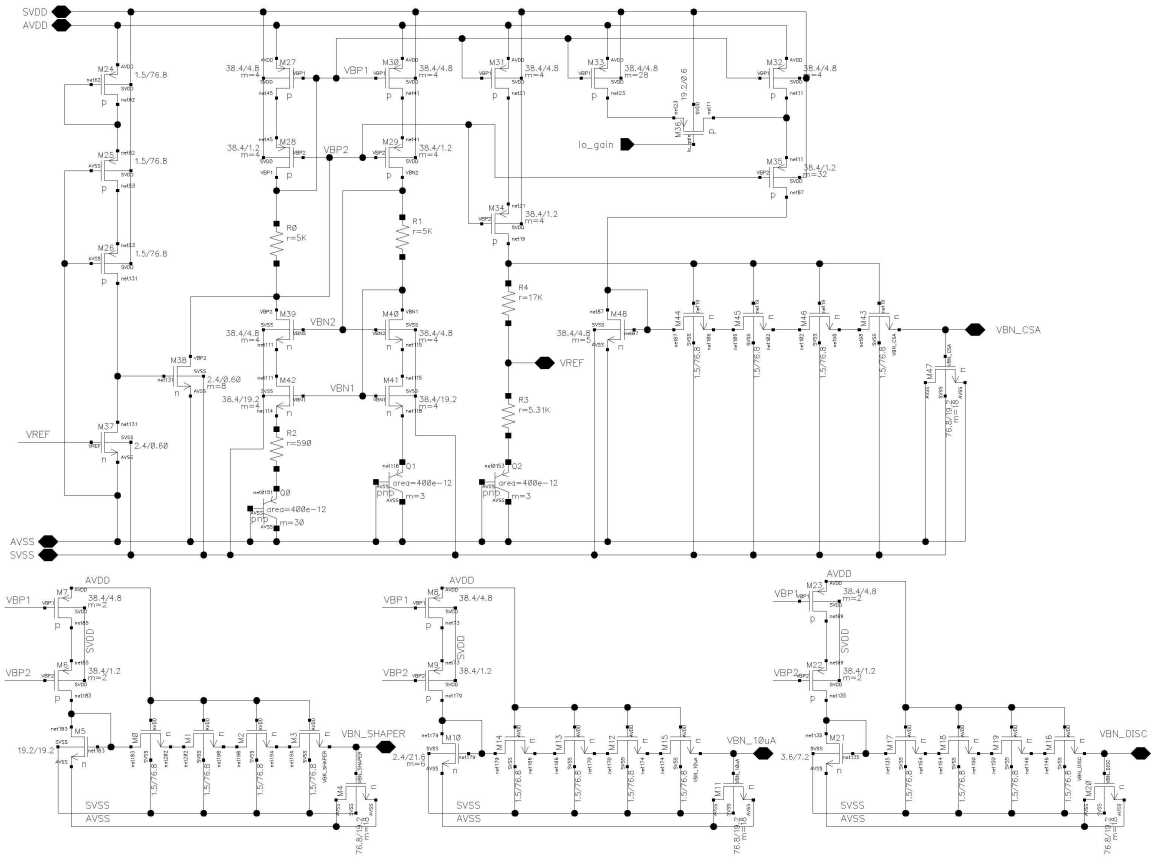


Figure 2.28 Schematic of bandgap voltage circuit.

2.11.1.2 Constant current source

The time-to-voltage converter requires a constant current to charge a capacitor, thereby, producing a voltage that varies linearly with time but independent of temperature or supply voltage. The schematic for the constant current is shown in Figure 2.29. It is important that the charging current display little, if any, temperature dependence. The bandgap voltage, described in the previous section, is applied across a temperature independent resistor. The temperature independence of the resistance is accomplished through a series combination of a resistance with a positive temperature coefficient (ny polysilicon) and a resistance with a negative temperature coefficient (hy polysilicon).This circuit produces two different bias voltages depending on which mode the TVC is currently operating.

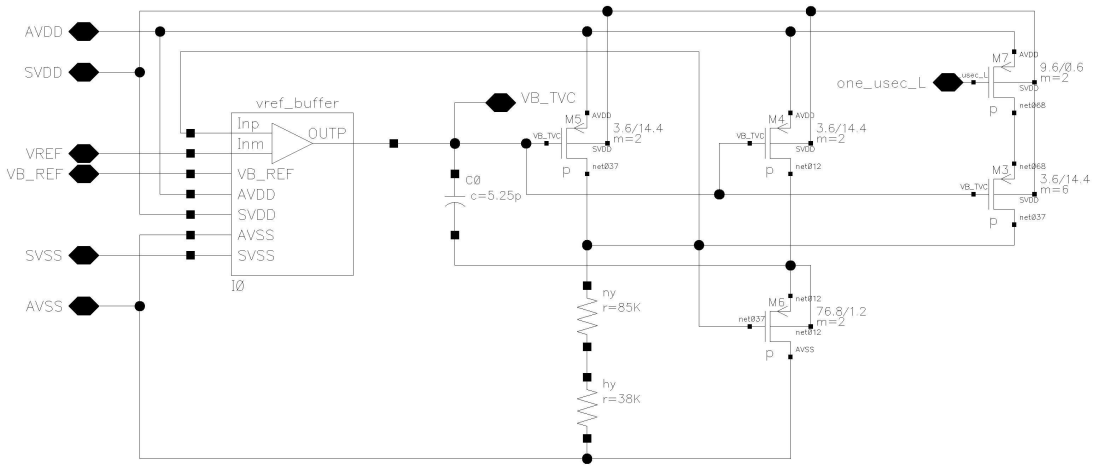


Figure 2.29 Schematic of constant current circuit.

2.11.1.3 DAC voltage reference

The bandgap reference is also used as a reference for the DACs that are used to offset compensate the leading edge discriminators. The 1.233 Volt reference is used in a feedback

loop along with a 60 KΩ resistor to generate a 20 µA current. This cicuit also provides the bias voltage required for generating the multiplicity output.

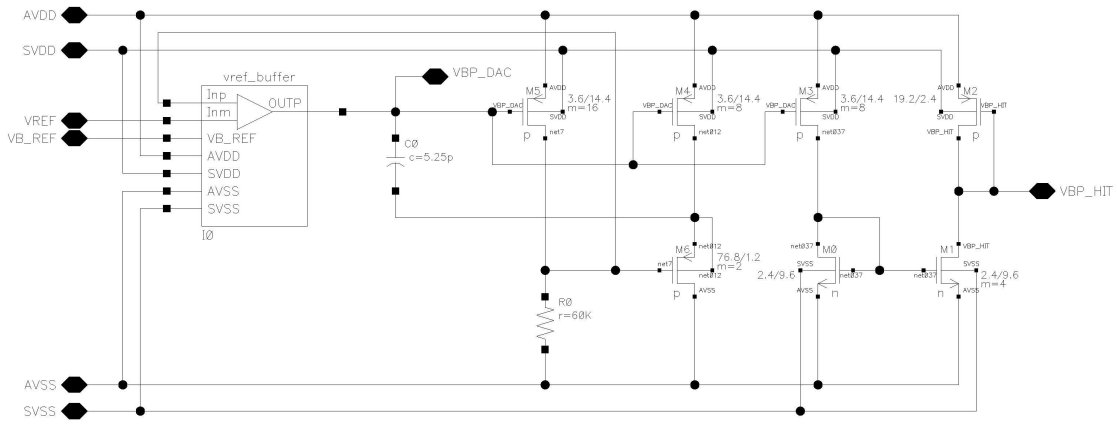


Figure 2.30 Schematic of DAC bias circuit.

2.11.2 Digital circuits

The digital circuits that provides the proper control signals for the channels in the IC and that contains the read out circuits is shown in Figure 2.31.

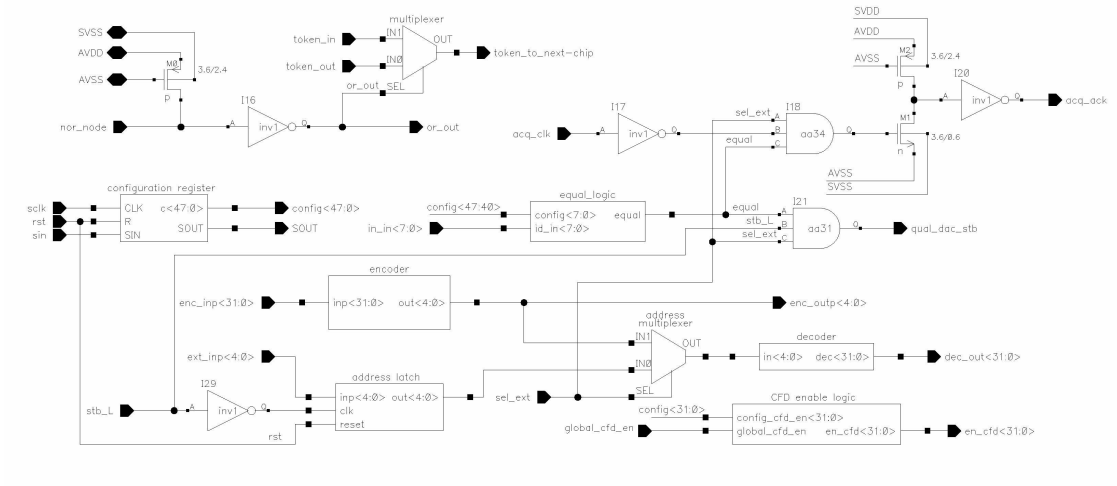


Figure 2.31 Block diagram of the common digital circuitry.

2.11.2.1 Configuration register

The configuration register is a serial shift register that is 48 bits (6 bytes) long. Bit 47 should be loaded first and bit 0 is loaded last. Data is applied to the input *sin*. Shifting occurs on the rising edge of register clock (*sclk*) and therefore *sin* data must be stable and valid on each rising edge of *sclk*. Data emerges from the configuration register at the output, *sout*. A positive pulse on the input *dig_rst* will reset all bits of the configuration register to 0.

The configuration register bit assignments along with the default state after a digital reset has been performed is shown in Table 2.1.

Bit Position	Function	Default
0	0 = Enable CFD Ch 0. 1 = Disable CFD Ch 0.	Ch 0 CFD enabled
1	0 = Enable CFD Ch 1. 1 = Disable CFD Ch 1.	Ch 1 CFD enabled
⋮	⋮	⋮
15	0 = Enable CFD Ch 15. 1 = Disable CFD Ch 15.	Ch 15 CFD enabled
16 – 31	Currently unused.	All bits 0
32	0 = Positive pulses at CSA out. 1 = Negative pulses at CSA out.	Negative pulses
33	0 = 1 µsec TVC range. 1 = 250 nsec TVC range.	1 µsec range
34	0 = CSA high-gain mode. 1 = CSA low-gain mode.	High-gain mode
35	0 = test mode 1 OFF. 1 = test mode 1 ON i.e. CSA and shaper outputs for selected channel brought out to pins.	CSA and shaper not brought out to pins
36	0 = Enable internal CSA. 1 = Select external preamp.	Use internal CSA
37	0 = test mode 3 OFF. 1 = test mode 3 ON Peak sampling circuit of selected channel driven by external signal.	Test mode 3 OFF
38 – 39	Currently unused.	All bits 0
40 - 47	Bit 47 MS bit (8 bit ID).	Chip ID = 0

Table 2.1 Configuration register bit assignments

2.11.2.2 Encoder

The encoder shown in Figure 2.31 is a 32 – 5 encoder. The output is a binary code that indicates which of the 32 input lines is active (HIGH) *i.e.* which of the 32 channels is currently in need of attention. If none of the 32 inputs are active, the output code is "00000". This is not a priority encoder. If more than one input is high, the output code is unknown. In normal operation, it is not possible for more than one input to be high simultaneously. One line from each of the 16 channels is routed to the 32-5 encoder that is located within the common digital block within the special common channel. In the current 16 channel IC, only inputs 0 – 15 are used. Inputs 16 – 31 are connected to ground.

2.11.2.3 Decoder

The decoder shown in Figure 2.31 is a 5 – 32 decoder. The 5 bit input address determines which of the 32 output lines is active, thereby selecting one of the 32 channels. One (and only one) of the 32 lines is always "hot". The decoder outputs are the channel select lines. One of the 32 channels can be selected by applying the appropriate 5 bit channel code to this decoder. In the current 16 channel IC, outputs 16 – 31 are not connected.

2.11.2.4 Address multiplexer

The multiplexer shown in Figure 2.31 is a 2-1 multiplexer. It is used to multiplex two 5 bit addresses. One 5-bit address is the output of the 32 – 5 encoder described above. The other 5 bit code is an externally generated address. The sel_ext_addr signal selects the externally generated address.

2.11.2.5 Address latch

It is possible to latch the externally applied address and this is done using the address latch shown in Figure 2.31. This allows the external address lines to be used as data lines for the digital-to-analog (DAC) converters that are used for offset compensation in the leading edge discriminators.

The address latch is transparent when the signal dac_stb is low. On the rising edges of dac_stb, the external address is latched. Data intended for the DACs converters will be latched on the falling edge of dac_stb. Care must be taken to make sure that the delays are such that the data is latched into the DAC register before the channel select lines change.

2.11.2.6 Equal logic circuit

The equal logic circuit used in the common digital circuit is shown in Figure 2.32. This circuit compares two eight bit values and if they are the same, the output of the circuit goes high. The output signal (equal) is used as the strobe line to the DACs. So this circuit allows us to program the DAC specified by the external address, only when this address matches with the internal address specified in the configuration register.

Figure 2.32 Block diagram of the equal logic circuit.

2.11.2.7 Enable CFD circuit

The enable CFD block shown in Figure 2.31 consists of 32 two input logic AND

gates. The outputs from this block enable or disable the CFD circuits in all the 16 channels. In the current 16 channel IC, outputs 16 – 31 are not connected.

2.11.2.8 Status circuits

The common digital block also consists of circuits that can report the current status of the channels in the IC. The IC can notify the external world that at least one of the channel hit registers has been set. This is accomplished by ORing the outputs of the 16 hit registers. The OR function is distributed across the channels. The pulldown transistors reside in the respective channels. There is one common pull-up device along with an inverter located in the common digital block. The or_out output signal in Figure 2.31, when high indicates that at least one channel on the IC has been hit.

In a similar manner, the acq_ack signal indicates whether an acquisition is currently in progress. The falling edge of acq_ack signal, signals the end of data acquisition. Also, when no channel in the IC is been hit, it skips the acquisition process in that IC and signals the IC connected next to it to starts its acquisition process by making the token_to_next_chip signal low. This speeds up the acquisition process.

2.11.3 Source Followers

The common channel consists of both the p-type and the n-type source followers. Figure 2.33 shows the circuits of the source followers used. These source followers act as buffers. The p-type source followers are used to buffer the multiplicity output, TVC output and the positive peak output signals. The n-type source follower is used to buffer the negative peak output signal.

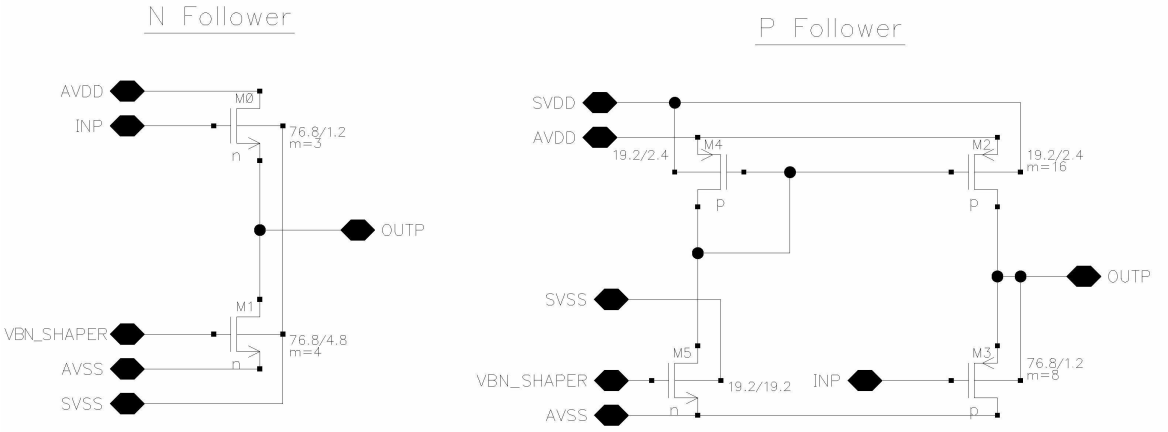


Figure 2.33 P-type and N-type source followers

CHAPTER 3

SIMULATED PERFORMANCE OF HINP16C

3.1 Charge Sensitive Amplifier (CSA)

3.1.1 Transfer Characteristics

The response of the CSA in high gain mode is illustrated in Figure 3.1. The simulation was performed with the value of the detector capacitance being 75pF and the CSA ground voltage set to 2.5V. The input to the system was a current pulse of width 80 ps and amplitude 10mA (equivalent to 5 million electrons). The output of the CSA is an exponentially decaying pulse with a rise time of 54 ns and decay time of 55 us. The peak amplitude of the waveform was 254 mV, but the expected value was 320 mV. The discrepancy (-2 dB) between the actual and the expected value of the peak is primarily due to the loss associated with the NFET source follower (bulk modulation effects) used to buffer the CSA output. A voltage of 1 volt at the output corresponds to 19 million electrons.

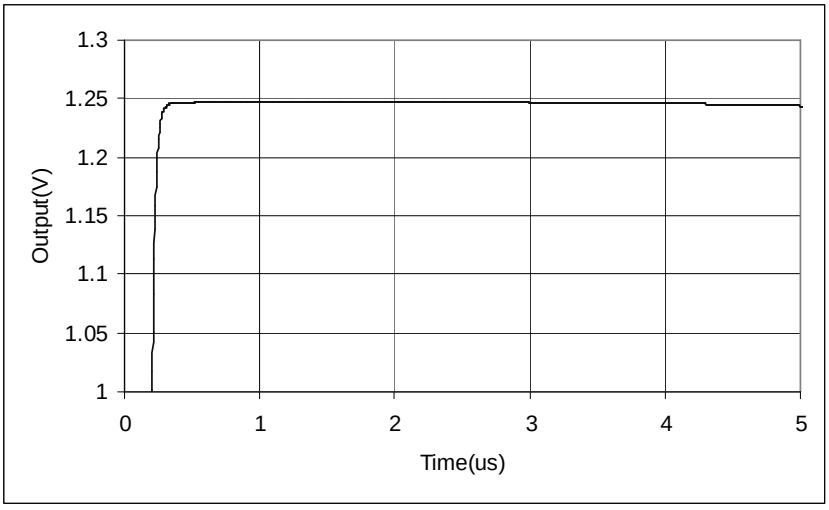


Figure 3.1 CSA output (high-gain mode) with C_{det} = 75 pF.

With a detector capacitance of 25 pF, the response begins to deteriorate, displaying ringing. For comparison purposes, the response at 15 pF is presented in Figure 3.2. It has a rise time of 9 ns. At 10 pF, the ringing is severe and not acceptable.

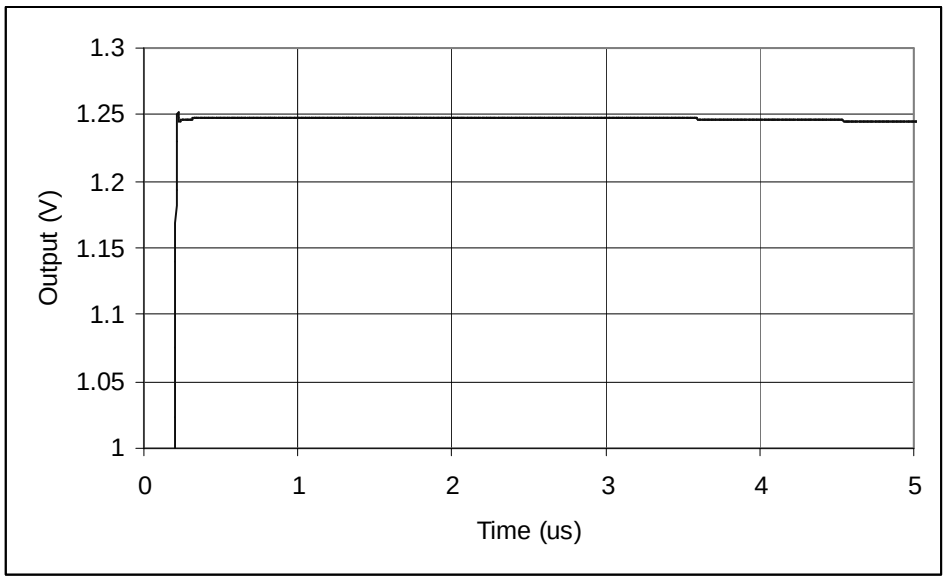


Figure 3.2 CSA output (high-gain mode) with C_{det} = 15 pF.

The transfer characteristics of the CSA in low gain mode looks similar to that of high gain mode, but 1V at the output corresponds to approximately 81 million electrons.

3.1.2 Linearity

The CSA is linear for both positive and negative going pulses. For the positive going output pulses, the CSA_GND voltage was 2.5 Volts. CSA_GND was held at 4 Volts when the CSA produced negative going output pulses. It is important that CSA_GND be set accordingly to achieve the maximum range over which the responses remain linear. The characterization was performed in both the high-gain and the low gain mode. The characterization for the high gain mode is illustrated in Figure 3.3. It can be seen that in high gain mode the CSA is linear up to 100Mev.

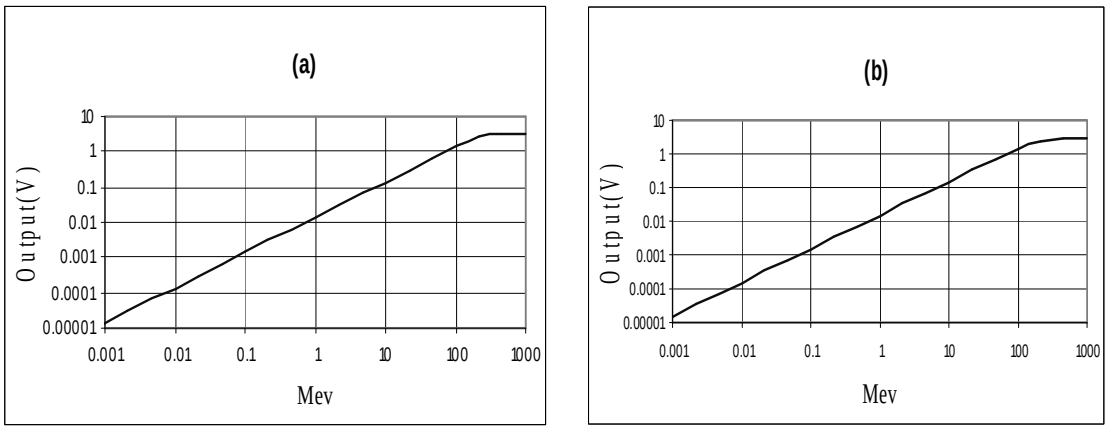


Figure3.3 Linearity of the CSA in high gain mode (a) positive pulses (b) negative pulses.

The characterization for the low gain mode is illustrated in Figure 3.4. In the low gain mode the CSA is linear up to 500 Mev.

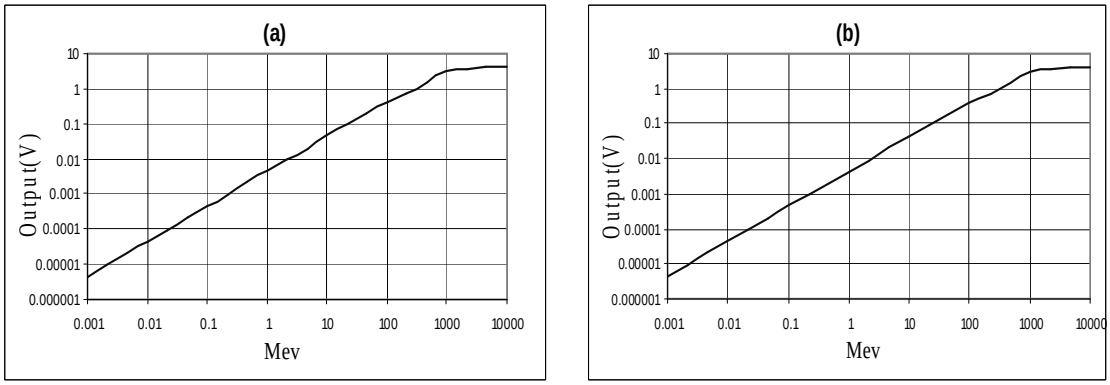


Figure 3.4 Linearity of the CSA in low gain mode (a) positive pulses (b) negative pulses.

3.1.3 Noise performance

The noise performance of the CSA is determined as a function of detector capacitance. The graphs represented in this section are the total integrated noise in a bandwidth of 10 KHz to 1 MHz which is roughly the bandwidth of the pulse shaper. The noise performance of the CSA in low gain mode is shown in Figure 3.5. The slope of the curve is approximately 15 electrons per pF. The noise at 0pF is 2475 electrons.

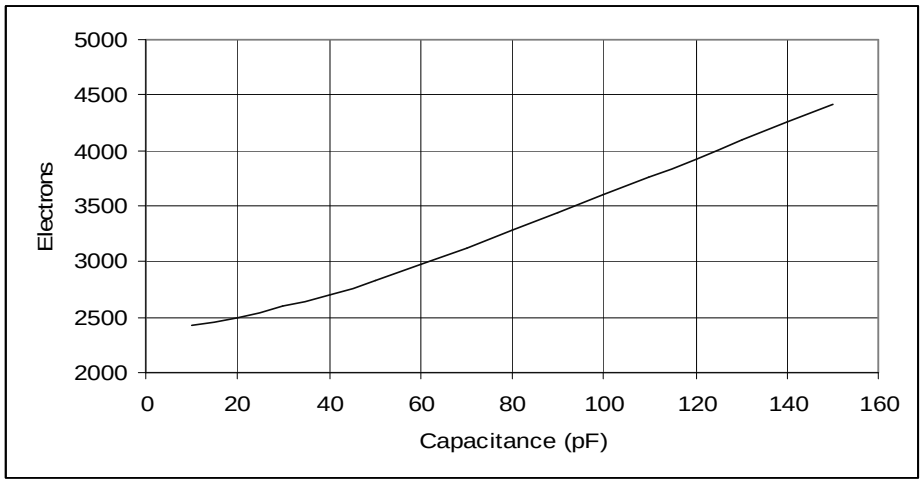


Figure 3.5 Noise performance of the CSA in low gain mode.

The noise performance of the CSA in high gain mode is shown in Figure 3.6. The slope of the curve is approximately 11 electrons per pF. The noise at 0pF is approximately 500 electrons. It can be seen from the graphs that the noise performance of the system improves with the decrease in the value of the detector capacitor. In the previous section it was proven that when the detector capacitance is less than 30 pF the CSA begins to oscillate. Therefore the value of the detector capacitor was chosen carefully to obtain the maximum throughput.

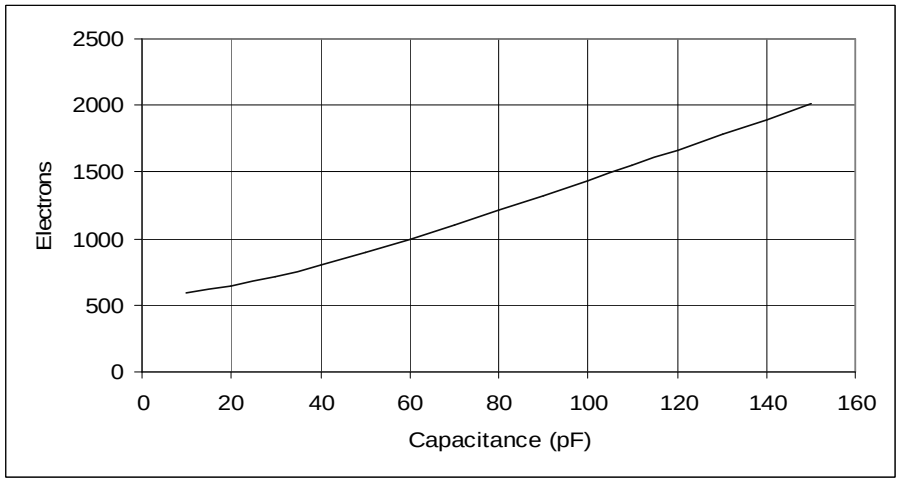


Figure 3.6 Noise performance of the CSA in high gain mode.

3.2 Pulse Shaper

3.2.1 Transfer Characteristics

Figure 3.7 shows the output of a pulse shaper for an input equivalent to 5 million electrons. This simulation was run with a control voltage, V_c of 2.5 volts. The peaking time for both the positive and negative pulses is approximately 1.2μs, return to baseline (1% of the peak amplitude) is approximately 2.8μs and the peak amplitude is 120mV. The shaper output returns to 0.1% of its peak value shortly after it returns to its baseline. Simulations were performed to ensure the smoothness of the semi-Gaussian pulse over the expected range of inputs. The simulations also revealed that for a fixed value of $V_{c,}$ the peaking time and the return to baseline time remained approximately the same.

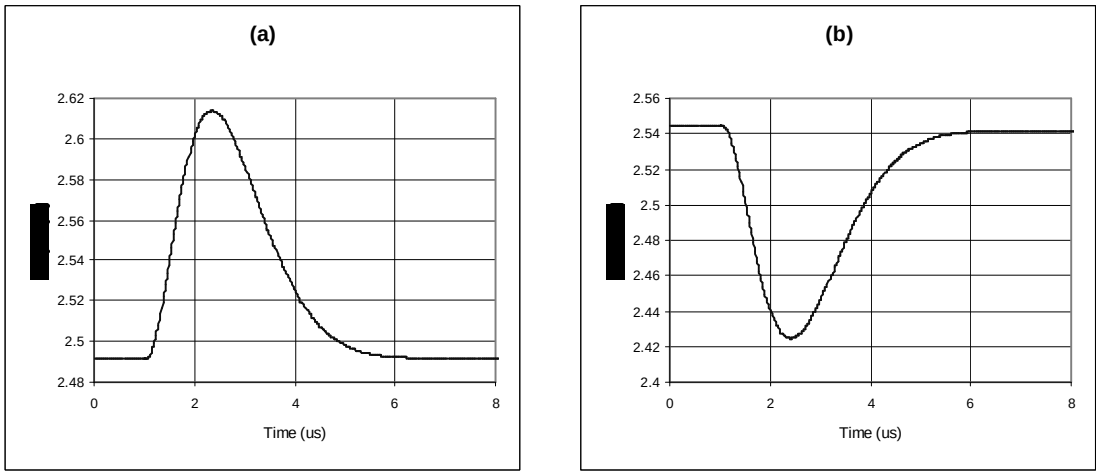


Figure 3.7 Shaper outputs of (a) positive pulses (b) negative pulses.

3.2.2 Peaking Time

The peaking time, return to baseline time and the peak amplitude of a shaper can be varied by changing the control voltage, V_c applied to the linear transconductors. The plot between the control V_c and peaking time for both the positive and negative pulses are shown

in Figure 3.8 (a) and (b) respectively. It reveals that the peaking time of the positive pulses of the shaper is 1µs when V_c is around 2.6 V and for the negative pulses when V_c is around 2.3 V. These plots reveal that the peaking time is inversely proportional the control voltage, V_c.

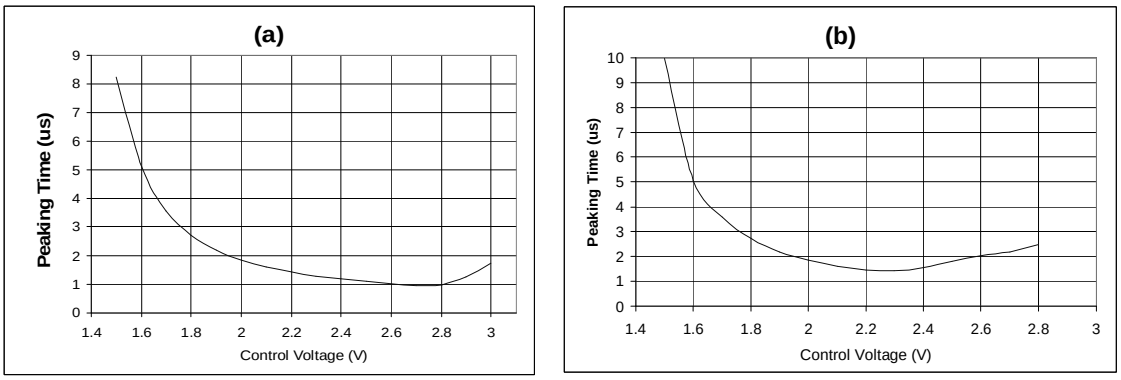


Figure3.8 Peaking time Verses Control voltage (a) positive pulses (b) negative pulses.

Figure 3.9 (a) and (b) shows the plots between the peaking time and the peak amplitude for positive and negative pulses respectively. It can be noted that the peak amplitude varies inversely with the control voltage between 1.7 V to 2.7 V.

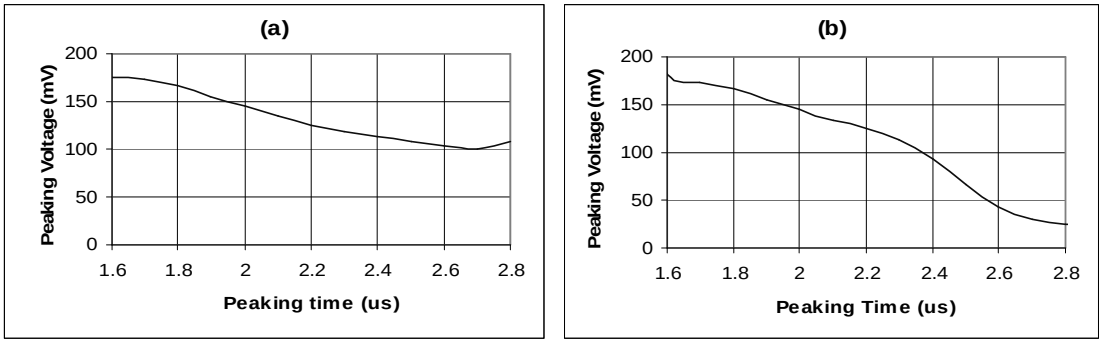


Figure 3.9 Peak time Verses Peak voltage (a) positive pulses (b) negative pulses.

3.2.3 Linearity

The linearity of the shaper for positive and negative pulses is shown in Figure 3.10 (a) and (b) respectively. For both the positive and the negative pulses the control voltage, V_c was

maintained at 2.5 V. It can be seen from the plots that the shaper is linear close to 100 Mev for both the pulses.

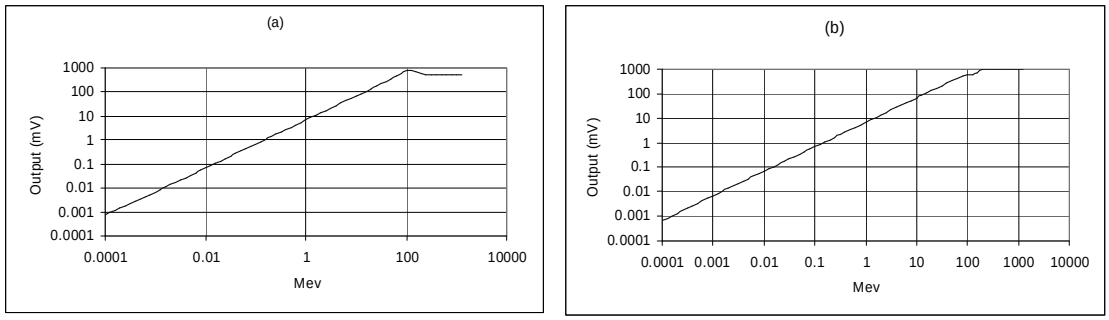


Figure 3.10 Linearity of shaper (a) positive pulses (b) negative pulses.

3.2.4 Noise Performance

Noise analysis of shaper was performed and it was found that the total integrated noise in the shaper to be approximately a constant for all values of the detector capacitance at the input of CSA. The noise at the output of the shaper is approximately 2400 electrons. It can be observed that for higher values of detector capacitance the noise performance is limited by the CSA.

3.3 Peak Sampler Linearity

The linearity of the peak sampler is mainly dependent on the linearity of the gain amplifier stage. The linearity of the gain amplifier for positive and the negative pulses are shown in Figure 3.11. The gain amplifier has a gain of 4 (12 dB). It can be seen from the plots for both the positive and the negative pulses that the gain amplifier is highly linear up to 80 Mev. The linearity of the gain amplifier can be increased by either reducing the gain of the amplifier or by increasing the supply voltage to the amplifier.

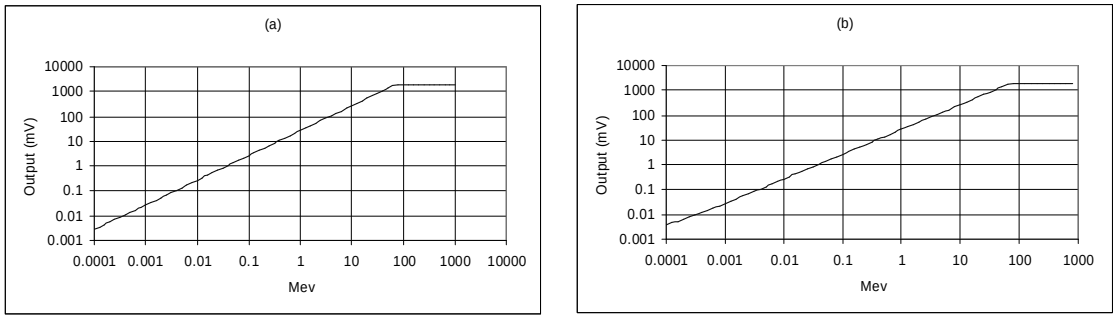


Figure 3.11 Linearity of gain amplifier (a) positive pulses (b) negative pulses

The linearity of the positive and the negative pulse sampling circuit is shown in Figure 3.12 (a) and (b) respectively. It is highly linear between 1 Mev and 80 Mev. The slight non linearity at the lower end is due to some DC offset introduced by the peak sampling circuit. If the offset is constant for a wide range of the input it can be easily corrected.

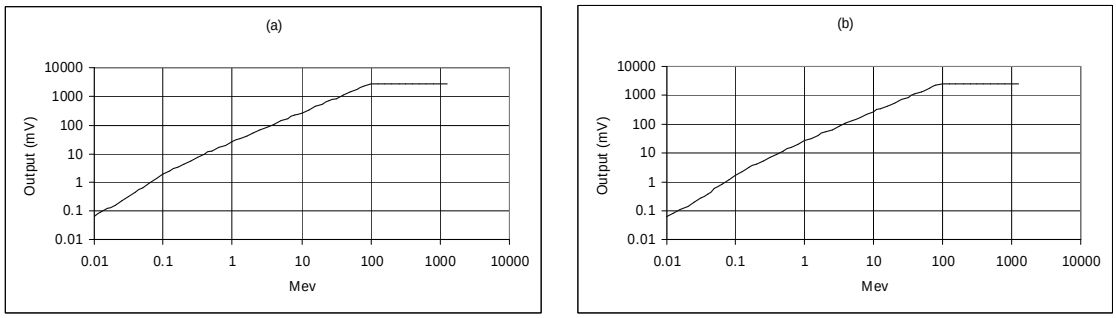


Figure 3.12 Linearity of peak sampler (a) positive pulses (b) negative pulses

3.4 Pseudo Constant Fraction Discriminator Walk

The pseudo constant fraction discriminator (CFD) displays very good walk characteristics. Walk is the variation of the measured arrival time of a signal due to the variations in the amplitude of the signal and can be caused by signal risetime or the nonideal properties of the discriminator. The propagation delay through the discriminator varies as a function of signal overdrive, signal underdrive and slope of the signal as it crosses the threshold.

The walk plot of the CFD is illustrated in Figure 3.13. The CFD was driven with exponential pulses. The risetime was 50 nsec and the decay time was 55 μsec. The pulse amplitude was varied from 0.5mV to 1.5V, corresponding to detector inputs ranging from 9500 electrons to 28.5 million electrons. The simulator maximum time step was specified as 100 ps. Some of the variation in propagation delay can be attributed to this relatively large time step. It can be seen from the graph that the walk in the CFD is less than 500 ps over a range of 50 dB.

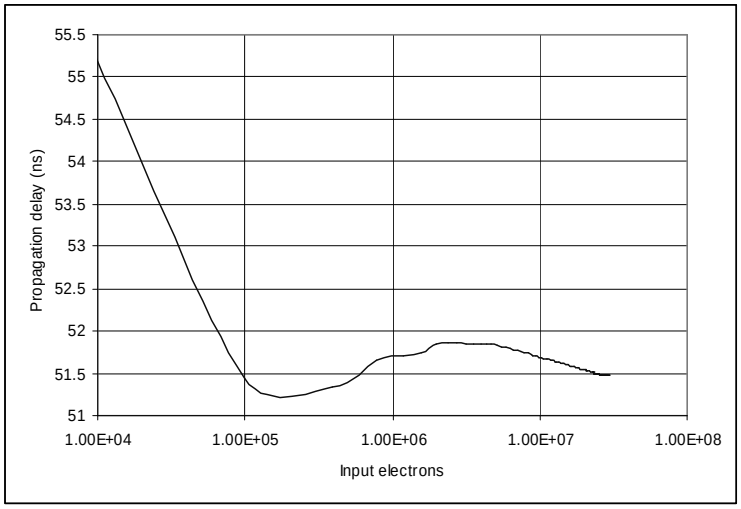


Figure 3.13 Walk plot of the CFD.

3.5 Time to Voltage Converter

The transfer function for the time-to-voltage converter (250 nsec range) is illustrated in Figure 3.14. The slope of the curve below is 3.31 mV / ns. The DC offset is approximately 1Volt and represents the threshold voltage of the source follower buffer. The range does not extend to time t = 0. The range over which the curves linearity is characterized is from time t = 3 ns to t = 250 ns. The integral nonlinearity is 435 ps. If a

range starting at time t = 2 ns is used, the integral nonlinearity increases to 570 ps. If the range starting at time t=10 ns is used, the integral nonlinearity is approximately 50 ps.

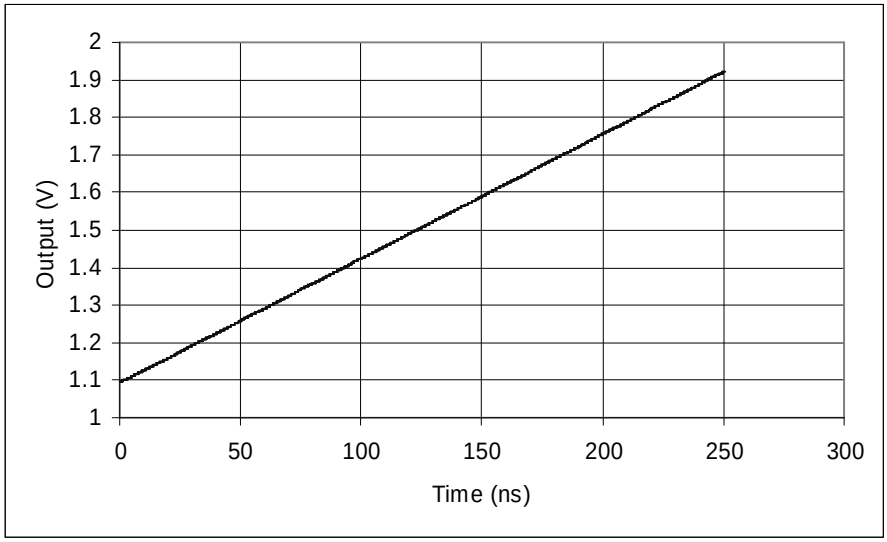


Figure 3.14 Linearity of the TVC in 250ns range.

The transfer function for the time-to-voltage converter in 1 µsec range is illustrated in Figure 3.15. The slope of the curve is 0.833mv/ns. The integral non linearity between time t = 2 ns to t = 1 µs is 2.8 ns. If a range starting at time t = 10 ns is used, the integral nonlinearity decreases to 1.3 ns. If the range starting at time t=30 ns is used, the integral nonlinearity is approximately 125 ps.

Simulations were done on the TVC under various temperatures and slope of the curves were the same in all the cases. Figure 3.16 illustrates that the idling point of the TVC changes with respect to temperature. The slope of the curve is -1.465 mV/ ºC. The idling point of the TVC should be monitored at regular basis to avoid any timing errors during the acquisitions of the data.

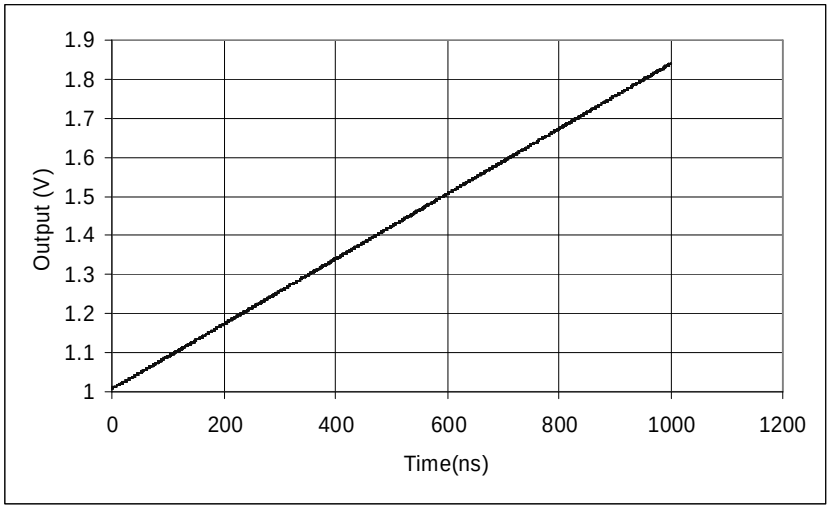


Figure 3.15 Linearity of the TVC in 1μs range.

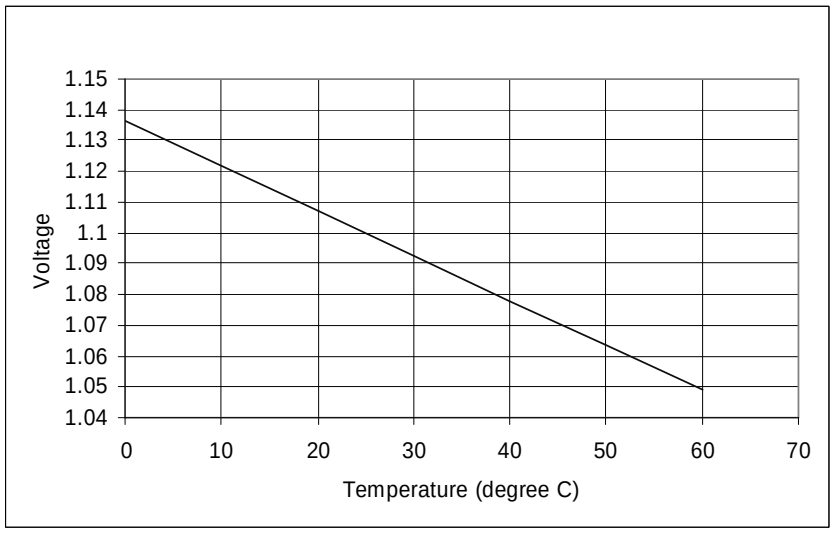


Figure 3.16 Idling point plot of the TVC in 250 ns range.

3.6 Analog Reset Logic

The one shot in the analog reset circuit produces a negative-going pulse with a variable pulse width. The delay, common to all channels on the IC, can be varied by controlling the control voltage (DLY_VC). Figure 3.17 below illustrates that the delay can be reliably varied from a few hundred nano-seconds to around 100 μsec by varying the

control voltage between 1 Volt and 4 Volts. A control voltage of 2.5 Volts yields a delay of 1 μsec which is approximately equal to the peaking time of the shaper.

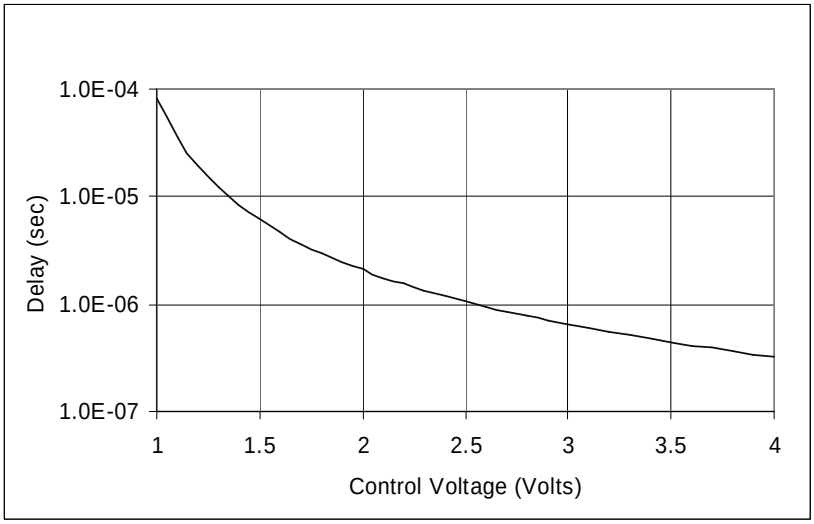


Figure 3.17 Plot of variable one shot delay versus control voltage.

3.7 Bias Circuits

3.7.1 Bandgap voltage reference

The transient response of the bandgap circuit is shown in Figure 3.18. It can be seen from the plot that the bandgap voltage starts up correctly when the supply voltage reaches 5 volts independent of the risetime of the supply voltage. In Figure 3.18(a) the risetime for the supply voltage is 100μs and in (b) the risetime is 1 mV.

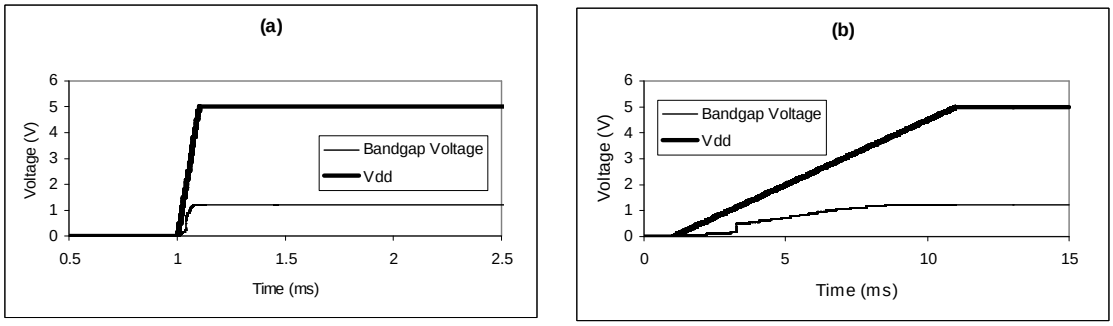


Figure 3.18 Transient response of the bandgap circuit.

The performance of the bandgap circuit is summarized in Table 3.1 and the summary of bias voltages and associated currents at 27ºC is given in Table 3.2.

Supply current (typical):	1.5 mA (high-gain mode) 0.75 mA (low-gain mode)
Noise at Vref output:	100 µV (10 Hz - 10 MHz)
Nominal Output:	1.233 Volts (27 ºC)
Temperature coefficient:	+65 µV / ºC

Table 3.1 Performance of bandgap circuit

Bandgap voltage Bandgap current	1.2331 Volts 1.2106E-01 mA
CSA: VBN_CSA I_CSA	 1.3694 Volts in high gain mode 0.91164 in low gain mode 9.0598E-01 mA
Shaper: VBN_SHAPER I_SHAPER	 1.8519 Volts 6.0567E-02 mA
CFD: VBN_DISC I_DISC VBN_10µA I_10µA VBP1_DAC VBP_HIT	 2.6245 Volts 6.0551E-02 mA 1.9829 Volts 1.0094E-02 mA 3.3796 Volts 3.4597 Volts
TVC: VBN_TVC	 3.3650 Volts
Peak Sampler: VBN_SHAPER I_SHAPER	 1.8519 Volts 6.0567E-02 mA

Table 3.2 Summary of bias voltages and associated currents at 27ºC

Variations of the bandgap output with respect to temperature are illustrated in Figure 3.19. The bandgap voltage varies only by 5 mV between the normal operating range of 0ºC and 40ºC.

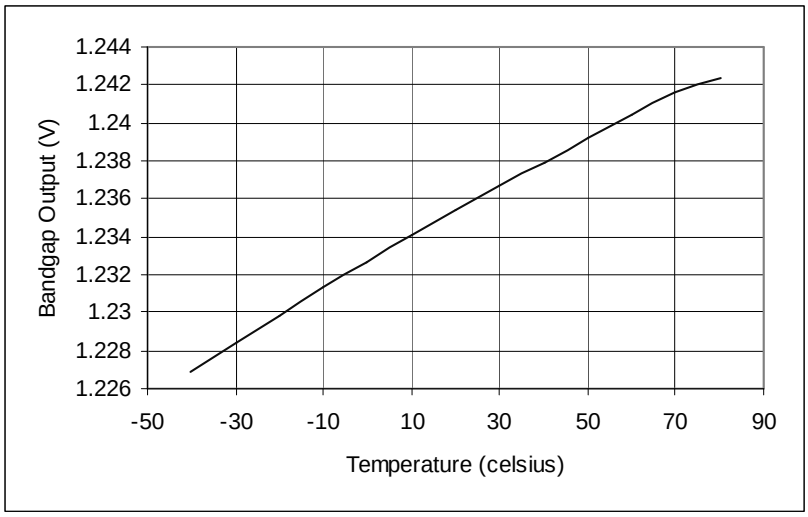


Figure 3.19 Temperature dependence of the bandgap voltage.

3.7.2 Constant current source

The constant current source is used in the TVC circuit to produce a constant current to charge the capacitor. Variations of the current with respect to temperature are illustrated in Figure 3.20. The Performance of constant current source is summarized below:

Nominal Output: 10.038 uA (27 ºC)

Temperature coefficient: + 850 pA / ºC

Current varies by +/- 0.17% around the nominal temperature of 27 ºC. Since the capacitor has virtually a zero temperature coefficient, this will also be the relative error in time measurements. We conclude by comparing Figure 3.19 and Figure 3.20 that the temperature dependence of the current source is almost completely due to the temperature dependence of the bandgap voltage.

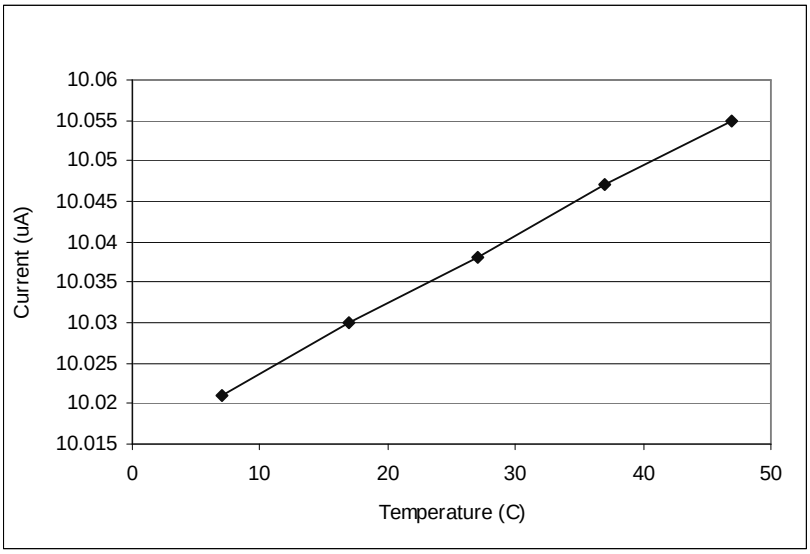


Figure 3.20 Temperature dependence of the current source

3.8 Source Followers

The source followers used as buffers in the common channel is shown in Figure 3.21. It can be seen that both n-type and p-type source followers are linear upto 3.5 volts at its input.

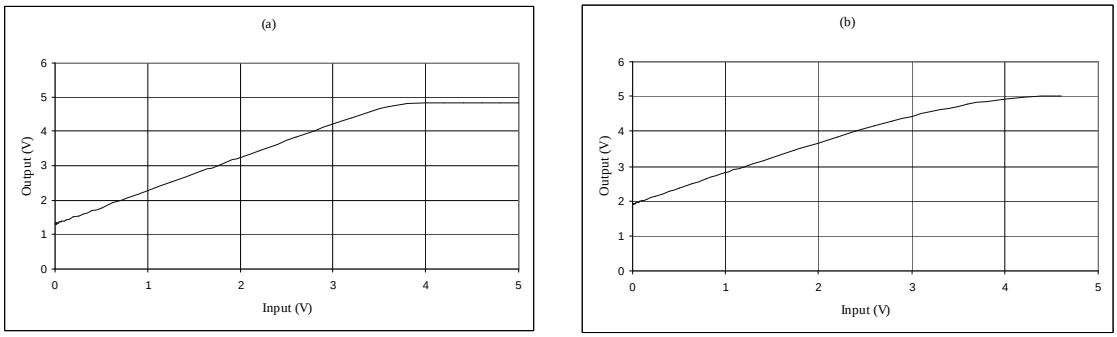


Figure 3.21 Linearity of source follower (a) p-type (b) n-type

3.9 Macro Model Simulation

The macro model simulations were performed on all the 16 channels to check the correctness of the design. To perform these simulations each block in the design was replaced with the corresponding veriloga codes. To generate the required control signals for the design unit, a testbench was written.

Figure 3.22 shows the outputs of the major analog units (single channel) present in the design: the CSA, pulse shaper, peak sampler, CFD and the TVC. Figure 3.23

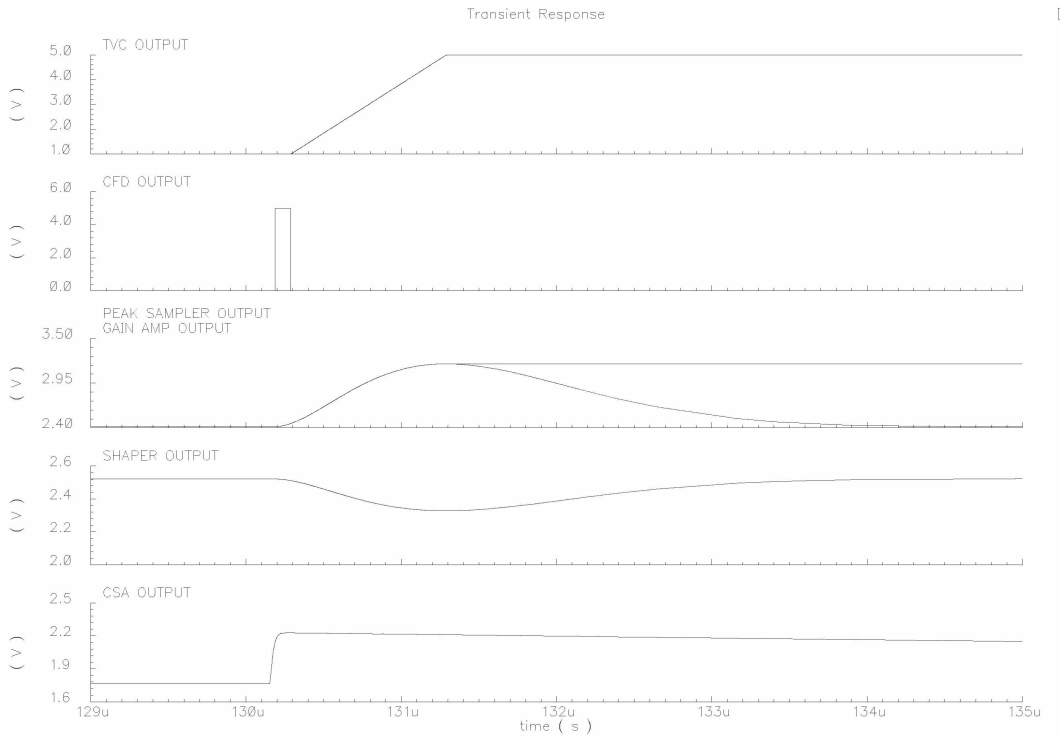


Figure 3.22 Output waveforms of the major analog units

illustrates the operation of the CFD. It can be seen from the figure that the leading edge discriminator fires before the zero crossing discriminator. The narrow pulse is generated through the outputs from both the discriminators. The CFD output is a 100 ns pulse generated by the passing the narrow pulse through a one shot circuit in the CFD.

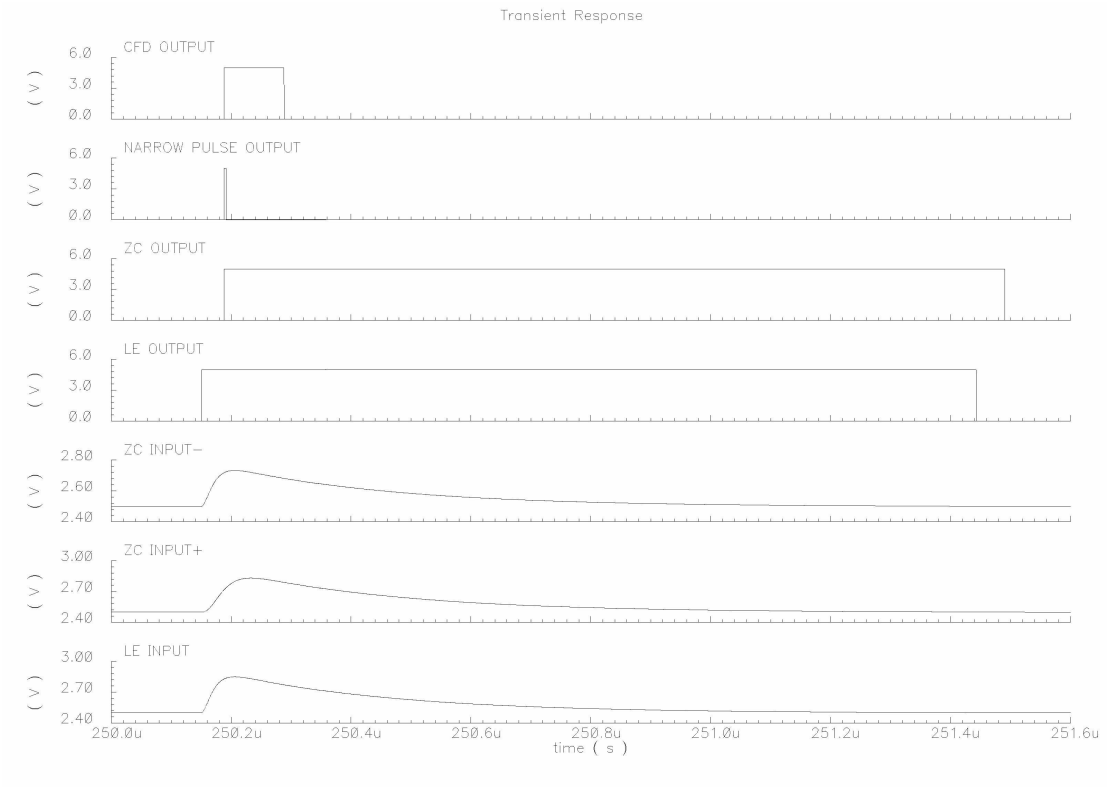


Figure 3.23 Waveforms in the CFD

Figure 3.24 shows the output of the CFD when the DAC is programmed. It can be seen that when the voltage at the output of the DAC is greater than the input to the leading edge discriminator, the CFD does not fire. Different threshold can be set for different channels in the IC. To do this each channel's DAC should be programmed individually.

The peak sampler is designed to operate in two modes. In the first mode, it tracks the peak only when the CFD fires and in the second it can be forced to track the peak even when the CFD has not fired. It can be seen in Figure 3.24 that when the force track signal is logic high, the peak sampler tracks the peak even when the CFD has not fired. Also, when the force track is low it does not track the peak.

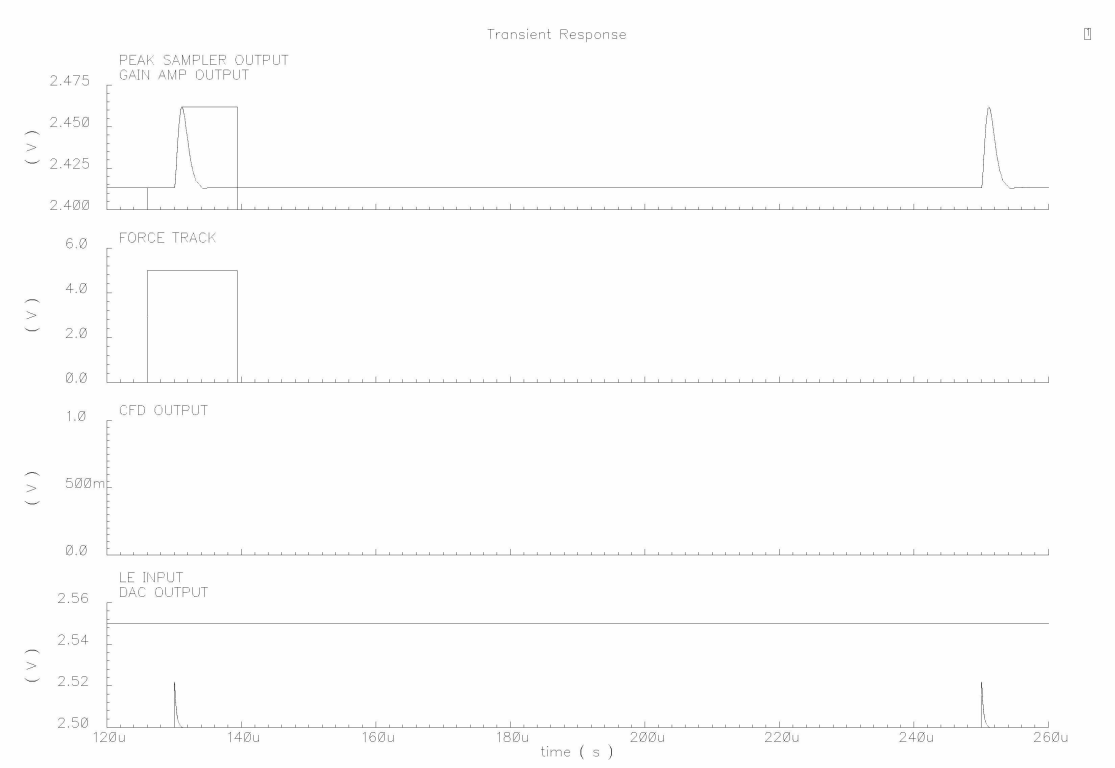


Figure 3.24 Waveforms in the CFD when DAC is programmed.

The sequence of events occurring in the hit logic circuit in the CFD is a bit complex. Figure 3.25 shows the different signals that can clear the hit register. The only signal that can set the hit register is the output of the CFD. There are three signals that can reset the hit register. The first signal is the force track signal which forces the hit register to be cleared. The second signal is the automatic reset of the analog reset signal. The third signal is the falling edge of the acquisition clock for that particular channel during the process of data acquisition.

Figure 3.26 shows the multiplicity and the OR output in the common circuits. The multiplicity output is a measure of the number of channels that are hit. It can be seen that 15 out of the possible 16 channels are hit. The multiplicity output is read off the chip during the

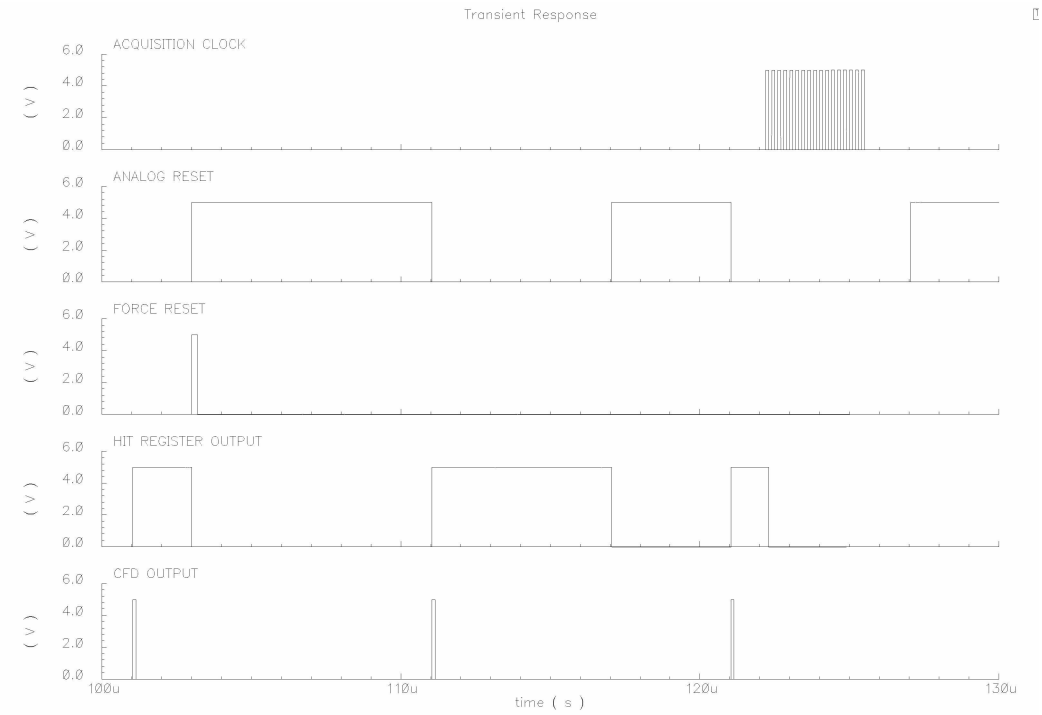


Figure 3.25 Waveforms of hit logic circuit in the CFD

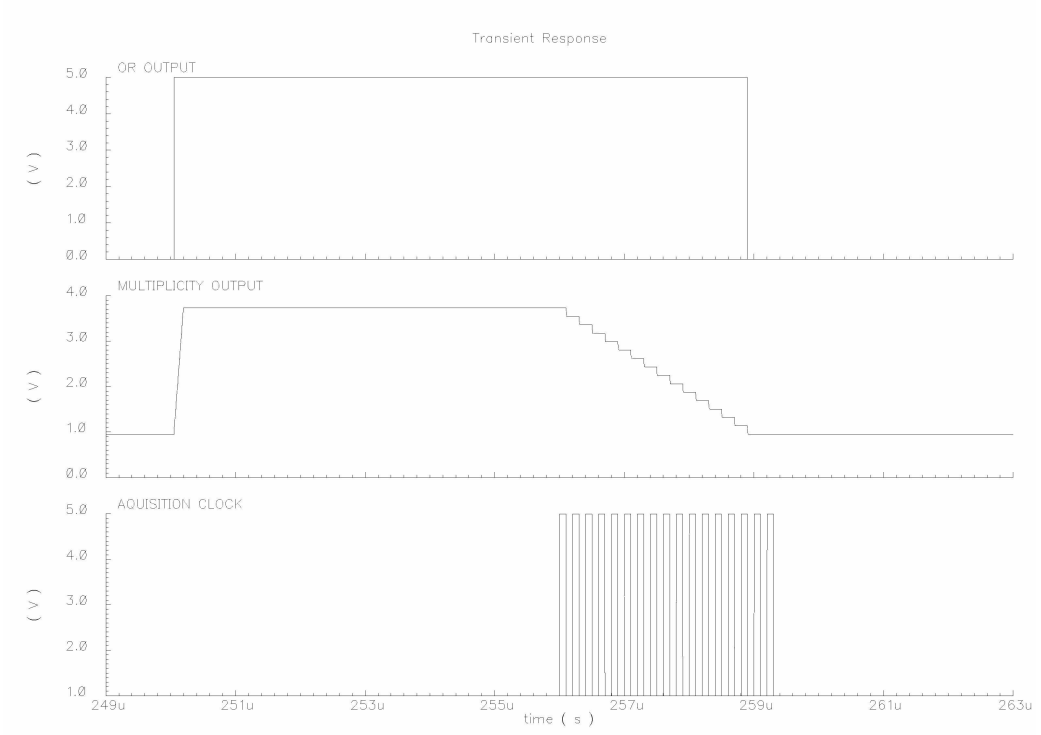


Figure 3.26 Waveforms showing multiplicity and OR outputs

acquisition process. The OR output which is an indication of at least one channel been hit, stays high until the data from the last channel that is hit has been read out.

Figure 3.27 demonstrates the working of the TVC. The output from the CFD starts the TVC and it keeps integrating till it encounters the common stop signal. The common stop signal is common for all the channels. By scanning the outputs of the TVC the timing information of the channels that are hit can be determined. The TVC is reset by the analog reset signal.

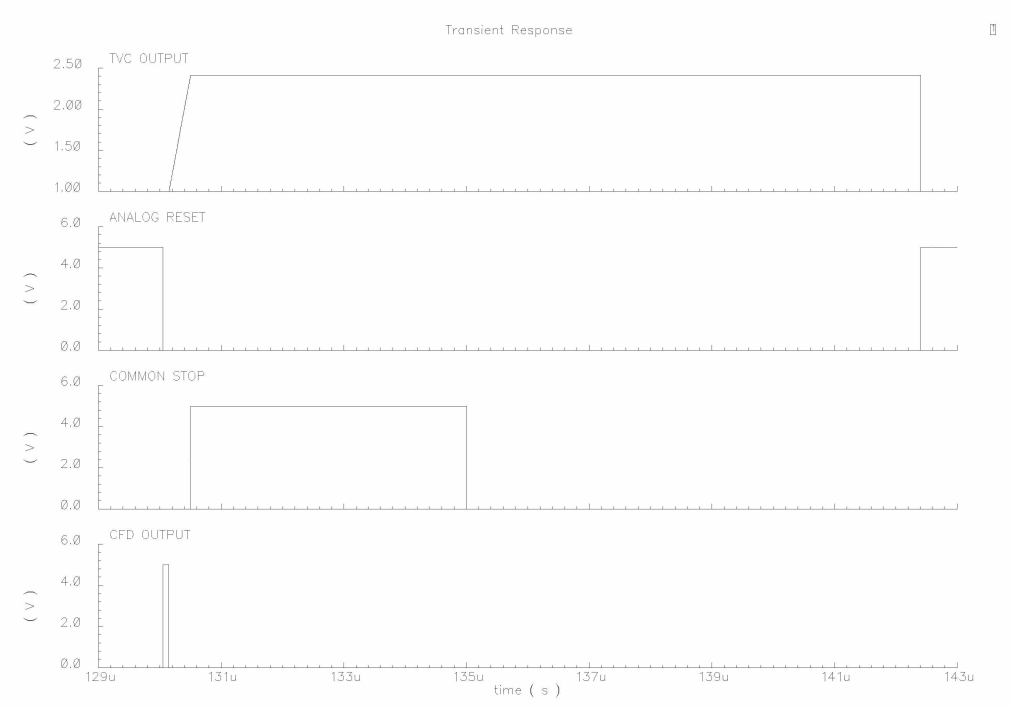


Figure 3.27 Waveforms showing the working of the TVC

Figure 3.28 shows the operation of the analog reset circuit. The analog reset signal can be brought out of reset in two different ways. It can be forced to come of reset by bringing the force reset signal high. The analog reset also comes out of reset automatically when it is timed out. The timing is controlled by the control voltage in the variable one-shot.

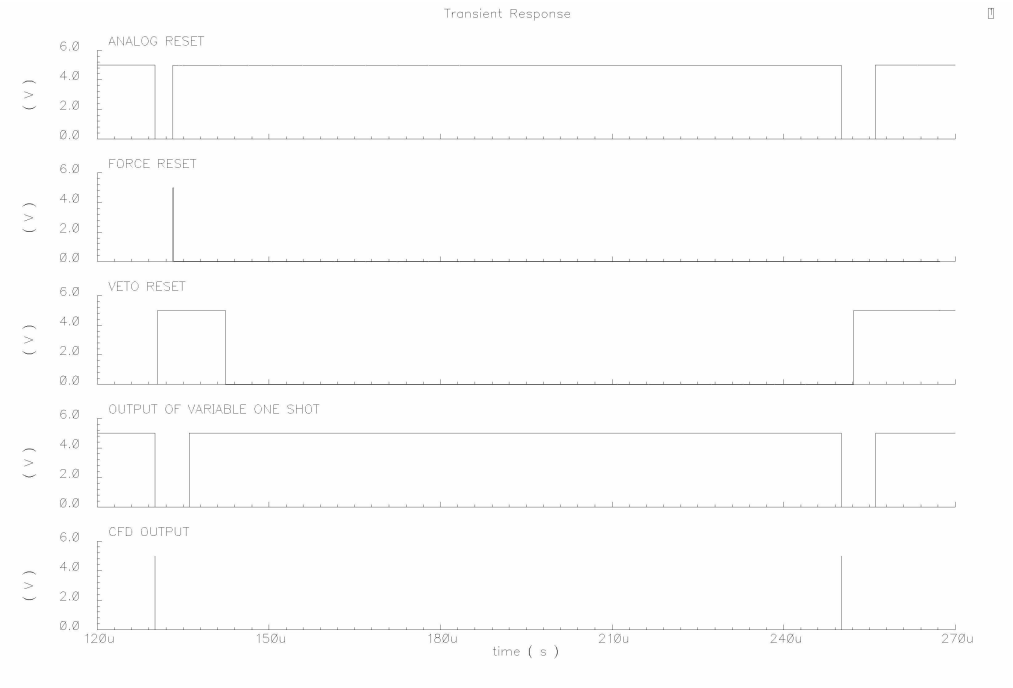


Figure 3.28 Waveforms showing the operation of the analog reset circuit

3.10 Power Dissipation

The Figure 3.29 shows the total power distribution among different block in the IC. The total current flowing through all the 16 channels and the common channel is around 107

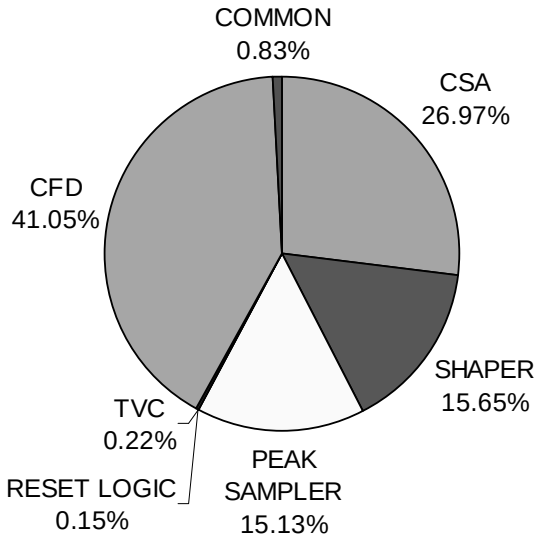


Figure 3.29 Power distribution chart

mA. Under normal conditions, the IC operates at 5 volts dissipating a total power of 536 mW. It can be seen that the CFD and the CSA dissipate most of the power in the chip.

3.11 Area Distribution

The Figure 3.30 shows the distribution in the area occupied by the individual blocks in the chip.

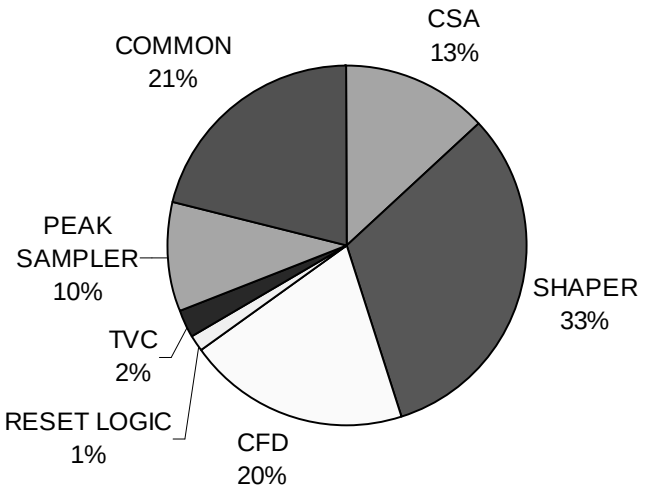


Figure 3.30 Area distribution chart

CHAPTER 4

HINP16C PERFORMANCE

The 16 channel integrated circuit that was fabricated in 0.5-micron double-poly, triple-metal CMOS nwell technology was successfully tested. The performance of the IC in the real world will be discussed in this chapter.

4.1 Charge Sensitive Amplifier

The charge sensitive amplifier, which is the input stage to the radiation sensor, was successfully tested for input of various amplitudes and polarities. When the IC was put in the test mode where the output from the CSA is sent to the pads, the rise time of the CSA was found to be around 200ns and the decay time was around 55 µs. But, the rise time calculated during the simulations was between 45-50 ns. The difference in the rise time arises due to the extra capacitance inserted by the analog output pad, which was unaccounted for during the simulations. Another notable reason for the increase in the rise time is the increase in the resistance of the switch that latches the CSA output to the output pad.

The decay time of the CSA is very close to the simulated value. The CSA was found to be linear for a wide range of inputs in both the gain modes. The CSA had the expected noise performance. Even when the CSA is pulsed through the pulser pads, it produces the desired output.

4.2 Pulse Shaper

The linearity and the noise performance of the pulse shaper agreed with the results

shown Chapter 3. The output of the shaper was a perfect semi-Gaussian pulse whose peaking time was controlled by the control voltage (V_c). The peaking time could be effectively controlled if the control voltage was between 1.5 and 2.75 volts. The output amplitude was approximately 40 % its input, which is a good indication the correct operation of the shaper.

4.3 Pseudo Constant Fraction Discriminator

The CFD was found to have some problems in generating a logic high output, which is an indication of that channel been hit. This problem in the CFD was traced back to the narrow pulse circuit and the latch present in the one shot circuit. When a channel was hit the CFD could not fire due to the asymmetrical loading at the output of the cross coupled latch built using NOR gates. This would not have been a major problem if the input pulse to the latch had been wide. The input to the latch was from the narrow pulse circuit which produces an output that has a pulse width of just 1.5 ns. This narrow pulse along with the asymmetrical loading at the output of the latch caused the CFD not to fire.

Figure 4.1 shows the output of the CFD when a channel is hit. It can be seen that the output of the CFD barely crosses the logic high threshold value. This is the output of the CFD when there is 250 fF more capacitive load due to the wiring capacitance at one of the outputs of the latch. There are two ways by which this problem can be solved. The first method is to increase the pulse width of the output from the narrow pulse circuit. And, the second method is to buffer the outputs of the latch.

This problem in the CFD can be corrected without sending the design again for fabrication with the above mentioned changes. The latch present in the existing IC is shown

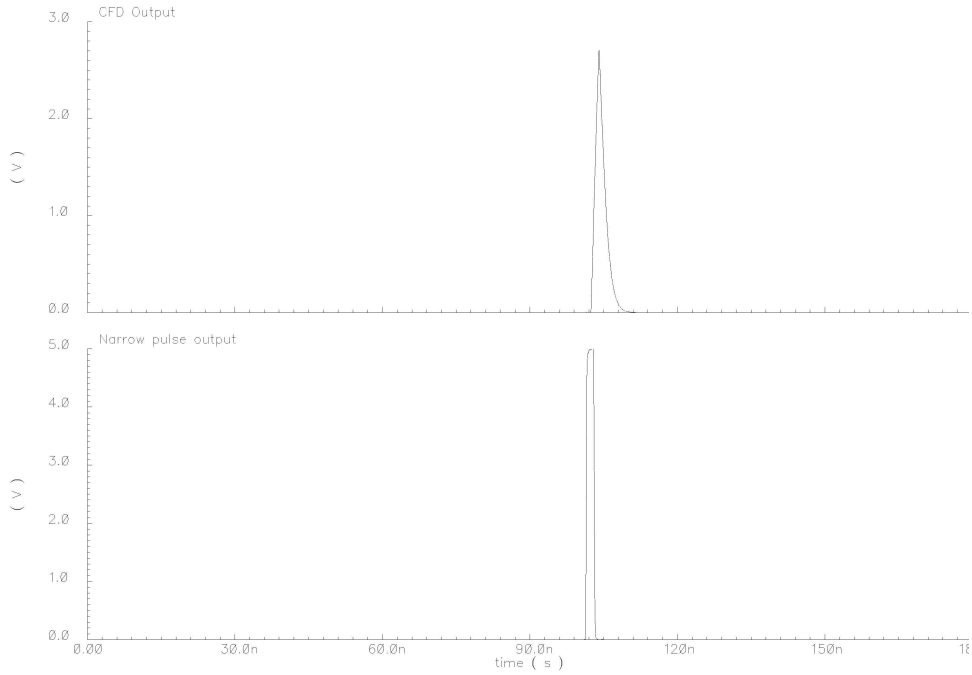


Figure 4.1 Output of the CFD due to asymmetric loading in the latch

in Figure 4.2 . The OR gate was built using a NOR gate and an inverter. The revised design is also shown in Figure 4.2, where the output fed back to the inputs from the outputs of the NOR gate present inside the OR gate. This change in the design adds two inverters and thus avoiding any asymmetrical loading at the output of the latch. This revised design can be implemented in the existing IC using the focused ion beam (FIB) technology. The FIB system is a widely utilized tool in IC device modification, semiconductor process monitoring, failure analysis and micromachining.

In the existing IC the hit register present in the hit logic circuit is not set because the CFD does not fire. This affects the OR output, which is an indication of at least one channel been hit. This will also affect the multiplicity output, is a measure of the number of channels that are hit.

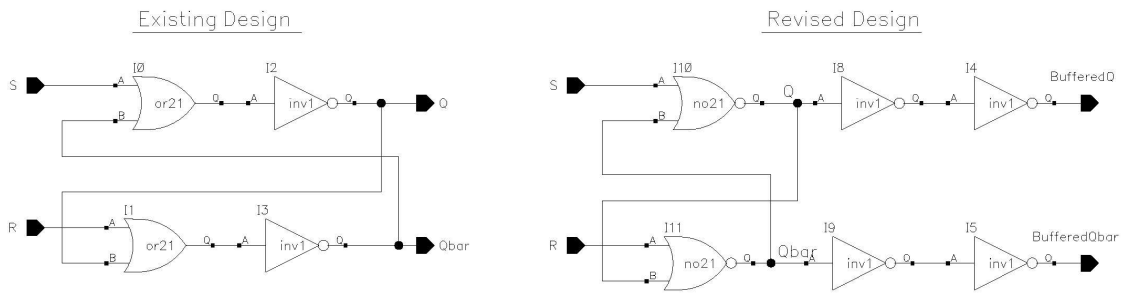


Figure 4.2 Existing and the revised design of the latch in the one shot

The DACs present in the CFD can be programmed successfully to cancel out the offsets present at the input to the leading edge discriminator and to set a threshold value for the input. The leading edge and the zero crossing discriminators are found to be operating as discussed in Chapter 3.

4.4 Peak Sampler

The peak sampler which outputs the energy information was found to be linear and possessed a good noise performance. The peak sampler tracks the peaks only if it is put in the force track mode. This is because, the signal that put it in track mode is the output from the hit register and since the CFD does not fire properly, the hit register does not get set. This is a problem in the CFD and not the peak sampler. As indicated by the simulations in Chapter 3, the linearity of the peak sampler can be improved by reducing the gain in the gain amplifier.

4.5 Time to-Voltage Converter

The TVC, which provides the timing information of the events occurring in the channel, works the way the simulations specify in Chapter 3. When the TVC is operating in the 250 ns range, the slope for the output versus time is 3.25 mv/ns which agree with the simulation results. The TVC does not start integrating if the CFD does not fire. This is due to

the fact that the CFD output starts the TVC. But if it does start it responds to the stop signal and holds the output for processing. The arrival of the reset signal from the analog reset circuit promptly resets the TVC.

4.6 Analog Reset Logic

The analog reset circuits responds to the output from the narrow pulse circuit in the CFD. This takes the analog reset signal out of reset. The pulse width of the analog reset signal was varied by adjusting the control voltage in the one shot present in the circuit. As discussed in Chapter 2 the circuit responds for both the force reset and the automatic reset signals.

4.7 Common Digital Circuits

The digital circuitry present in the common channel was successfully tested and was functioned as expected. The configuration register was loaded in 48 clock cycles and the outputs of each flipflop were able enabled/disabled the CFDs, set the address of the chip and perform all the other functions. The OR output and the multiplicity outputs agree with the simulation results in Chapter 3. The DACs could be programmed successfully by specifying an address and the DAC value.

4.8 Bias Circuits

The bandgap reference circuit in the bias circuitry starts perfectly when the power for the IC is turned on. If the bandgap voltage is not right none of the circuit in the chip would operate correctly. As the bandgap has started it assures that the constant current source and

the DAC bias circuits will also start correctly. This was confirmed from the results obtained from the TVC and the DAC. Also the power dissipated by the chip was around 600 mW, which is pretty close to the value predicted from the simulations. This also proves that the bias circuits in the common channel operate as desired.

CHAPTER 5

CONCLUSION AND FUTURE WORK

5.1 Conclusion

The IC for the use in low and intermediate energy nuclear physics experiments was successfully designed and implemented in 0.5-micron double poly, triple-metal CMOS nwell technology. The 16 channel IC detects the particles that strike the silicon detectors and process the data effectively. The minor problem that had occurred in the CFD circuit can be easily fixed by the FIB technology. When this minor problem has been fixed the IC will become fully functional. The oscillations that occurred in the previous chip that was introduced by the zero crossing detector was completely eradicated by introducing a DC feedback loop in to that circuit..

5.2 Future Work

The gain of the CSA needs to be increased to improve the linear range of the IC. This can be achieved by reducing the feedback capacitance in the CSA .If a value of 500 fF is used for the feedback capacitor then the value of the resistor needs to be increased 5 times the existing value of 10 MΩ. Building a resistor of 50 MΩ using the high resistance poly material is going to occupy a lot of the die area. So, an effective design that could provide the 50 MΩ resistance and that does not occupy a large area should be designed. Another factor that should be taken into account while designing the new CSA is that the noise performance should be comparable to the existing system. Finally, the chip should contain 32 channels instead of just 16 channels.

REFERENCES

[1] Z.Y Chang and W.M.C Sansen, Kluwer. "Low- Noise and Wide Band Amplifiers in Bipolar and CMOS Technologies", Kluwer Academic Publishers, 1991.

[2] Suharli Tedja, Jan Van der Spiegel, and Hugh H.Williams, "A CMOS Low-Noise and Low-Power Charge Sampling Integrated Circuits for Capacitive Detector/Sensor Interfaces", *IEEE Journal of Solid-State Circuits*, Vol.30, No.2, February 1995.

[3] Kenneth R.Laker and Willy M.C.Sansen, "Design of Analog Integrated Circuits and Systems", McGraw-Hill, Inc. 1994.

[4] J.C.Santiard, CERN and K.Marent, IMEC vzw, "The Gasiplex0.7-2 Integrated Front-end Analog Processor for the Hmpid and the Dimuon Spectrometer of Alice", Internal CERN manuscript obtained from Lee Sobotka, Washington University.

[5] Paul D.Walker and Michel M.Green, "A Tunable Pulse- Shaping Filter for Use in a nuclear Spectrometer System", *IEEE Journal of Solid-State Circuits*, Vol.31, No.2, June 1996.

[6] Eric J.Van der Zwan, Eric A.M.Klumperink, and Event Seevinck, "A CMOS OTA for HF Filters with programmable Transfer Function", IEEE Journal of Solid-State Circuits", Vol.26, No.11, November 1991.

[7] Trond Saether, Chung-Chih Hung, Zheng Qi, Mohammed Ismail, and Oddavar Aaserud, " High Speed, High Linearity CMOS Buffer Amplifier", IEEE Journal of Solid-State Circuits, Vol.31, No.2, February 1996.

[8] Paul R.Gray and Robert G.Meyer, "Analysis and Design of Analog Integrated Circuits", John Wiley & Sons, Inc. 1977.

[9] Micheal L.Simpson, Charles L.Britton, Alan L.Wintenberg, and Glenn R.Young, "An Integrated CMOS Time Interval Measurement System with Sub nanosecond Resolution for the WA-98 Calorimeter", *IEEE Journal of Solid- State Circuits*, Vol.32, No.2, February 1997.

[10] T.J.Paulus, EG&G Ortec, "Timing Electronics and Fast Timing Methods with Scintillations Detectors", *IEEE Transactions on Nuclear Science*, Vol.NS-32, No.3, June 1985.

[11] David M.Binkley, CTI PET Systems, Inc., "Performance of Non-Delay-Line Constant-Fraction Discriminator Timing Circuits", *IEEE Transactions on Nuclear Science*, Vol.41, No.4, August 1994.

[12] M.L.Simpson, C.L.Britton, A.l. Wintenberg and G.R.Young, Oak Ridge National Laboratory, "An Integrated, CMOS, Constant-Fraction Timing Discriminator for Multichannel Detector Systems", *IEEE Transactions on Nuclear Science*, Vol.42, No.4, August 1995.

[13] M.L.Simpson and G.R.Young, Oak Ridge National Laboratory, R.G.Jackson and M.Xu, University of Tenessee, Knoxville, "A Monolithic, Constant-Fraction Discriminator Using Distributed R-C Delay Line Shaping", *IEEE Transactions on Nuclear Science*, Vol.43, No.3, June 1996.

[14]B.T.Turko, Lawrence Berkeley Laboratory, R.C.Smith, Los Alamos National Laboratory, "A Precision Timing Discriminator for High Density Detector Systems", *IEEE Transactions on Nuclear Science*, Vol.39, No.5, October 1992.

[15] David M.Binkley, Michael L.Simpson, James M. Rochelle, "A Monolithic, 2 µm CMOS Constant-Fraction Discriminator for Moderate Time Resolution Systems", *IEEE Transactions of Nuclear Science*, Vol.38, No.6, December 1991.

[16] G.De Geronimo, P.O'Connor and J.Grosholz, "A CMOS Baseline Holder (BLH) for Readout ASICs".

[17] Ron Hogervorst, John P.Tero, Ruud G. H. Eschauzier, and Johan H. Huijsing, " A Compact Power-Efficient 3 V CMOS Rail-to-Rail Input/Output Operational Amplifier for VLSI Cell Libraries", *IEEE Journal of Solid State Circuits*, Vol.29, No.12, December 1994.

[18] Reference material given by Dr. Lee Sobotka, Professor, Radiochemistry Department, Washington University, Saint Louis.

[19] Mahadevan Ganesan, " A CMOS Low Noise IC for Capacitive Detector Interfaces", M.S. Thesis Report, Electrical Engineering, May 2000.

[20] Mohamedsha Malikansari, " A CMOS Low Noise IC for Use in Colliding Particle Experiments", M.S. Thesis Report, Electrical Engineering, October 2001.

APPENDIX A

Pinout of the HINP16C IC

Pin number: 1
Pin name: **cfd_out**

Pin type: Digital output

Description: This is the output (for the selected channel) of the 100 ns one-shot that is triggered by the narrow output pulse from the CFD. The CFD outputs from all 32 channels are multiplexed. This is the output of the multiplexer.

Pin number: 2
Pin name: **acq_clk**

Pin type: Digital input

Description: This is the clock signal used for acquisition. The rising edge of "acq_clk" causes the active register to be set in a channel whose "hit" register is set AND whose "token_in" is active i.e. LOW. The falling edge of "acq_clk" in turn causes the "hit" register to be cleared. This is turn will potentially allow the "token_out" of the channel to be active i.e. LOW; thereby, enabling the next channel in the chain. The next rising edge of "acq_clk" will clear the active register.

Pin number: 3
Pin name: **a0**

Pin type: Bidirectional

Description: This is external address line a0. When the "sel_ext_addr" pin is HIGH, this line will be a DIGITAL INPUT and is the least significant bit of the address of the channel the user wishes to select. When the "sel_ext_addr" pin is LOW, this line will be a DIGITAL OUTPUT and will be the least significant bit of the address of the channel that is currently in need of attention.

Pin number: 4
Pin name: **a1**

Pin type: Bidirectional

Description: This is external address line a1. See description for address line a0.

Pin number: 5
Pin name: **a2**

Pin type: Bidirectional

Description: This is external address line a2. See description for address line a0.

Pin number: 6
Pin name: **a3**

Pin type: Bidirectional

Description: This is external address line a3. See description for address line a0.

Pin number: 7
Pin name: **a4**

Pin type: Bidirectional

Description: This is external address line a4. See description for address line a0.

Pin number: 8
Pin name: **or_out**

Pin type: Digital output

Description: The “or_out” pin will be HIGH if any hit register on the chip is set. A LOW on this pin indicates that NONE of the “hit” registers is set.

Pin number: 9
Pin name: **acq_ack**

Pin type: Digital output

Description: The “acq_ack” pin will be HIGH during the acquisition process and will go LOW once all channels have been acquired.

Pin number: 10
Pin name: **token_in**

Pin type: Digital input

Description: This is the token into the chip. It is active LOW.

Pin number: 11
Pin name: **token_out**

Pin type: Bidirectional

Description: This is the token_out of the chip. It is active LOW. When the line is high, an acquisition is in progress.

Pin number: 12
Pin name: **veto_rst**

Pin type: Digital input

Description: After a channel has been hit, the time-to-voltage and peak sampling circuits as well as the hit and active registers will automatically be reset UNLESS “veto_rst” is asserted (HIGH). The “veto_rst” signal must be continued to be asserted until the time when the automatic reset would have taken place.

Pin number: 13
Pin name: **force_rst**

Pin type: Digital input

Description: A positive going pulse on this line will reset the time-to-voltage and peak sampling circuits as well as the hit and active registers in ALL channels.

Pin number: 14
Pin name: **common_stop**

Pin type: Digital input

Description: When HIGH, halts the time-to-voltage converter in every channel. The time-to-voltage conversions will STOP even if the start conversion signal is still asserted.

Pin number: 15
Pin name: **global_cfd_en**

Pin type: Digital input

Description: When LOW, the CFDs for all channels are DISABLED. When HIGH, a channel’s CFD will be enabled provided the corresponding CFD enable bit is a ‘0’ in the configuration register.

IMPORTANT NOTE: After the "global_cfd_en" line is made active, it important that a reset be forced. This is accomplished by applying a positive pulse to the “force_rst” pin.

Pin number: 16
Pin name: **DGND1**

Pin type: Digital supply pin. Return currents for digital I/O pads.

Description: Connect to circuit ground.

Pin number: 17
Pin name: **DVDD1**

Pin type: Digital supply pin. Powers digital I/O pads.

Description: Connect to +5VDC.

Pin number: 18
Pin name: **id0**

Pin type: Bidirectional.

Description: Bit 0 of the chip identification code. Least significant bit. When "sel_ext_addr" is HIGH, id0 is an *input*.

Pin number: 19
Pin name: **id1**

Pin type: Bidirectional.

Description: Bit 1 of the chip identification code. When "sel_ext_addr" is HIGH, id1 is an *input*.

Pin number: 20
Pin name: **id2**

Pin type: Bidirectional.

Description: Bit 2 of the chip identification code. When "sel_ext_addr" is HIGH, id2 is an *input*.

Pin number: 21
Pin name: **id3**

Pin type: Bidirectional.

Description: Bit 3 of the chip identification code. When "sel_ext_addr" is HIGH, id3 is an *input*.

Pin number: 22
Pin name: **id4**

Pin type: Bidirectional.

Description: Bit 4 of the chip identification code. When "sel_ext_addr" is HIGH, id4 is an *input*.

Pin number: 23
Pin name: **id5**

Pin type: Bidirectional.

Description: Bit 5 of the chip identification code. When "sel_ext_addr" is HIGH, id5 is an *input*.

Pin number: 24
Pin name: **id6**

Pin type: Bidirectional.

Description: Bit 6 of the chip identification code. When "sel_ext_addr" is HIGH, id6 is an *input*.

Pin number: 25
Pin name: **id7**

Pin type: Bidirectional.

Description: Bit 7 of the chip identification code. When "sel_ext_addr" is HIGH, id7 is an *input*.

Pin number: 26
Pin name: **sel_ext_addr**

Pin type: Digital input

Description: When HIGH, this signal selects the external address as input to the 5-32 decoder used for selecting one of the 32 channels. When HIGH, makes a0-a5 lines as well as id0-id7 lines *inputs*.

Pin number: 27
Pin name: **dac_stb**

Pin type: Digital input

Description: Data on the address pins (a0-a5) are latched into an internal address latch on the rising edge of dac_stb. When dac_stb is high, data on the ext_addr lines will alter the DAC output whose channel is select by the address stored in the internal

address latch. On the falling edge the data on the address pins will be latched into the DAC register.

IMPORTANT NOTE: Data on address lines a0-a5 must be stable and valid on both rising and falling edge of “dac_stb”.

Pin number: 28
Pin name: **dac_sgn**

Pin type: Digital input

Description: While the magnitude of the DAC value is placed on the bidirectional external address lines (a0-a5), the algebraic sign has a dedicated input pin

Pin number: 29
Pin name: **rst**

Pin type: Digital input

Description: Master reset. Resets all of the digital logic. All bits of the configuration register are cleared. All of the DAC registers on chip are also cleared.

Pin number: 30
Pin name: **acq_all**

Pin type: Digital input

Description: A positive-going pulse will set the “hit” register in each of the channels. This can be useful if one wants

to force the acquisition of all channels on chip. AT PRESENT THERE IS NO WAY TO SET THE HIT REGISTER IN ANY ONE CHANNEL (other than by having the CFD fires). Do we need to change this?????

Pin number: 31
Pin name: **sout**

Pin type: Digital output

Description: Serial output from 48-bit configuration register.

Pin number: 32
Pin name: **sin**

Pin type: Digital input

Description: Serial clock for 48-bit configuration register. Data on “sin” pin must be valid on rising edge of “sclk”.

Pin number: 33
Pin name: **sclk**

Pin type: Digital input

Description: Serial clock for 48-bit configuration register. Data on “sin” pin must be valid on rising edge sclk.

Pin number: 34
Pin name: **force_track**

Pin type: Digital input

Description: When active (HIGH) forces all peak sampling circuits into track mode even if CFDs have not fired. Force_track must be asserted 500 nsec prior to peaking of shaper circuits. Must be held active until all data has been acquired. Releasing force_track causes a reset of peak sampling circuit.

Pin number: 35
Pin name: **quiet**

Pin type: Digital input

Description: When active (HIGH) mutes analog portions of CFD circuits.

Pins 36: UNUSED

Pin number: 37
Pin name: **AVSS**

Pin type: Analog input

Description: Connect to circuit ground.

Pin number: 38
Pin name: **AVDD**

Pin type: Analog supply pin.

Description: Connect to +5VDC. Supplies power to shaper, peak sampling and full bias circuits.

Pin number: 39
Pin name: **EXT_SHAPER**

Pin type: Analog input

Description: Can be used to apply a simulated SHAPER signal to the PEAK SAMPLING circuit. For this to happen the configuration register bit 37 should be active high.

Pins 40-43: UNUSED

Pin number: 44
Pin name: **EVEN_PULSER**

Pin type: Analog input

Description: Pulser input for even channels (0.2, 4, 6, etc.)

Pin number: 45
Pin name: **ODD_PULSER**

Pin type: Analog input

Description: Pulser input for odd channels (1, 3, 5, 7, etc.)

Pin number: 46
Pin name: **PULSER_AVDD**

Pin type: Analog input

Description: Connect to +5VDC.

Pin number: 47
Pin name: **PULSER_AVSS**

Pin type: Analog input

Description: Connect to circuit ground.

Pin number: 48
Pin name: **CH_IN15**

Pin type: Analog input

Description: Channel 15 detector input

Pin number: 49
Pin name: **CH_IN14**

Pin type: Analog input

Description: Channel 14 detector input

Pin number: 50
Pin name: **CH_IN13**

Pin type: Analog input

Description: Channel 13 detector input

Pin number: 51
Pin name: **CH_IN12**

Pin type: Analog input

Description: Channel 12 detector input

Pin number: 52
Pin name: **CH_IN11**

Pin type: Analog input

Description: Channel 11 detector input

Pin number: 53
Pin name: **CH_IN10**

Pin type: Analog input

Description: Channel 10 detector input

Pin number: 54
Pin name: **CH_IN9**

Pin type: Analog input

Description: Channel 9 detector input

Pin number: 55
Pin name: **CH_IN8**

Pin type: Analog input

Description: Channel 8 detector input

Pin number: 56
Pin name: **CSA_GND**

Pin type: Analog input

Description: This is a signal ground for the CSA circuits. For negative-going pulses at output of CSA, connect to 4.0 VDC. If processing positive-going pulses, connect to 2.5 VDC.

Pin number: 57
Pin name: **CSA_AVSS**

Pin type: Analog supply pin

Description: Connect to circuit ground

Pin number: 58
Pin name: **CSA_AVDD**

Pin type: Analog supply pin

Description: Connect to +5VDC.

Pin number: 59
Pin name: **CH_IN7**

Pin type: Analog input

Description: Channel 7 detector input

Pin number: 60
Pin name: **CH_IN6**

Pin type: Analog input

Description: Channel 6 detector input

Pin number: 61
Pin name: **CH_IN5**

Pin type: Analog input

Description: Channel 5 detector input

Pin number: 62
Pin name: **CH_IN4**

Pin type: Analog input

Description: Channel 4 detector input

Pin number: 63
Pin name: **CH_IN3**

Pin type: Analog input

Description: Channel 3 detector input

Pin number: 64
Pin name: **CH_IN2**

Pin type: Analog input

Description: Channel 2 detector input

Pin number: 65
Pin name: **CH_IN1**

Pin type: Analog input

Description: Channel 1 detector input

Pin number: 66
Pin name: **CH_IN0**

Pin type: Analog input

Description: Channel 0 detector input

Pins 67-135: UNUSED

Pin number: 136
Pin name: **AVDD1**

Pin type: Analog supply pin. Supplies power for the above.

Description: Connect to +5VDC.

Pin number: 137
Pin name: **AVSS1**

Pin type: Analog supply pin. Return currents for channels 0-7 for TVC circuits

Description: Connect to circuit ground.

Pin number: 138
Pin name: **TVC_OUT**

Pin type: Analog output

Description: Analog output voltage proportional to the duration of time that has elapsed between the channel being hit (the one presently selected) and the assertion of the “common_stop” signal.

Pin number: 139
Pin name: **PEAK_OUT**

Pin type: Analog output

Description: Peak amplitude of shaper signal for the selected channel.

Pin number: 140
Pin name: **CSA_OUT**

Pin type: Analog output

Description: CSA output for the selected channel. Only available if the “test mode” bit in the configuration register is a ‘1’.

Pin number: 141
Pin name: **SHAPER_OUT**

Pin type: Analog output

Description: Shaper output for the selected channel. Only available if the “test mode” bit in the configuration register is a ‘1’.

Pin number: 142
Pin name: **DLY_VC**

Pin type: Analog input

Description: Control voltage that determines the time delay between a channel being hit and the automatic reset of the time-to-voltage converter, the peak sampling circuit, and the active and hit registers in that channel.

Pin number: 143
Pin name: **DGND**

Pin type: Digital supply pin for above.

Description: Connect to circuit ground.

Pin number: 144
Pin name: **DVDD**

Pin type: Digital supply pin for common digital circuits, CFDs, and reset logic

Description: Connect to +5VDC.

Pin number: 145
Pin name: **PEAK_TIME**

Pin type: Analog input

Description: Control voltage for shaper peaking time. Set to 2.5 VDC for a 1 □sec peaking time (approximate).

Pin number: 146
Pin name: **MULTIPLICITY**

Pin type: Analog output

Description: Analog output voltage proportional to the number of channels whose hit registers is currently set.

Pin number: 147
Pin name: **AGND**

Pin type: Analog input

Description: Analog ground (2.5 Volts)

Pin number: 148
Pin name: **TVC_CAP_GND**

Pin type: Analog input

Description: Connect to a clean circuit ground.

Pin number: 149
Pin name: **AVDD2**

Pin type: Analog supply pin. Supply for channels 8-15 TVC circuits.

Description: Connect to +5VDC.

Pin number: 150
Pin name: **AVSS2**

Pin type: Analog supply pin. Return currents for channels 8-15 TVC circuits.

Description: Connect to circuit ground.

Pin number: 151
Pin name: **SUBSTRATE**

Pin type: Analog input

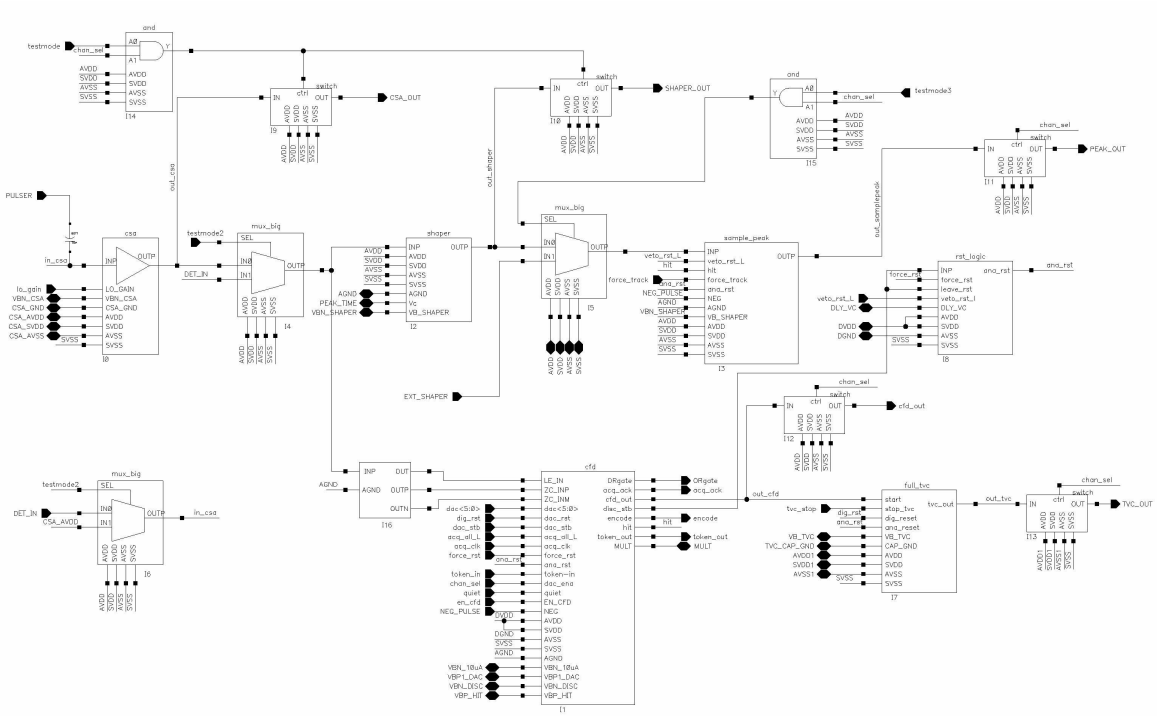
Description: Biases silicon substrate. Connect to clean circuit ground.

Pins 152-160: UNUSED

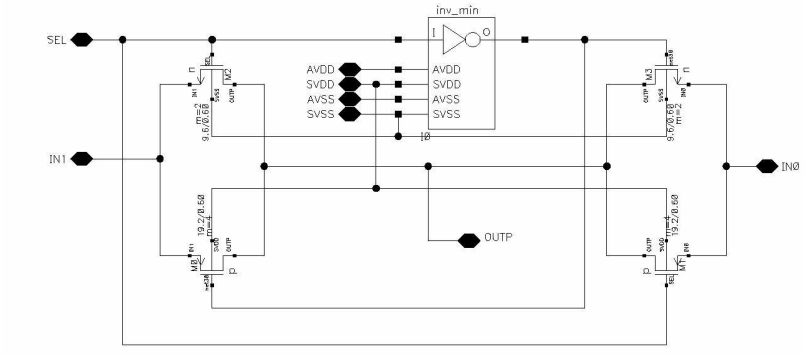
APPENDIX B

SCHEMATICS OF THE CIRCUITS USED IN THE IC

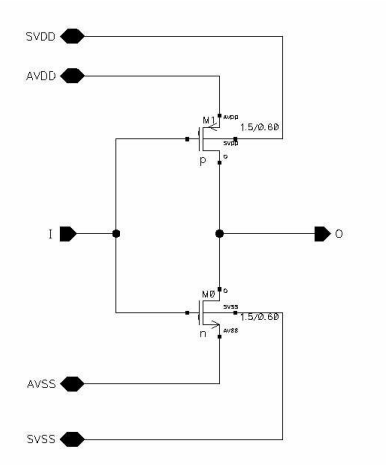
Block diagram of a single Channel



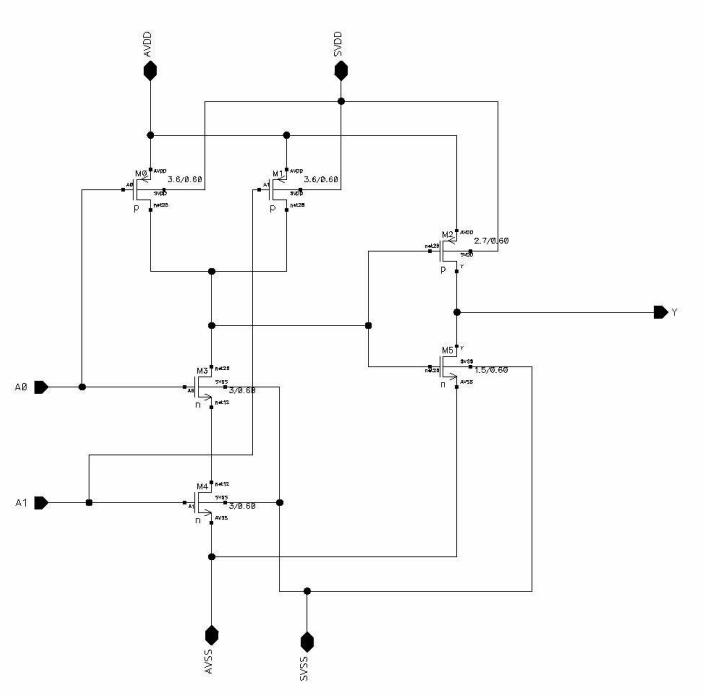
Schematic of the multiplexer used in Channel



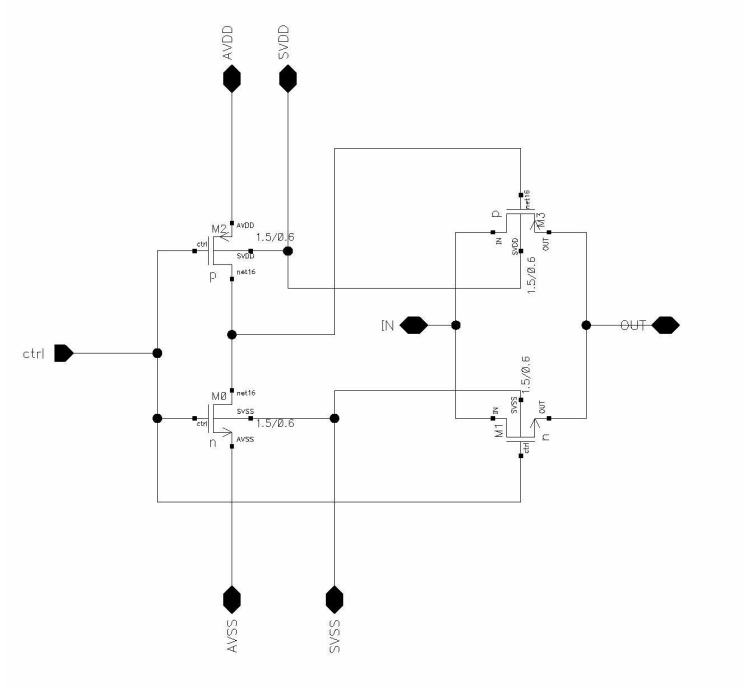
Schematic of the inverter used in the multiplexer



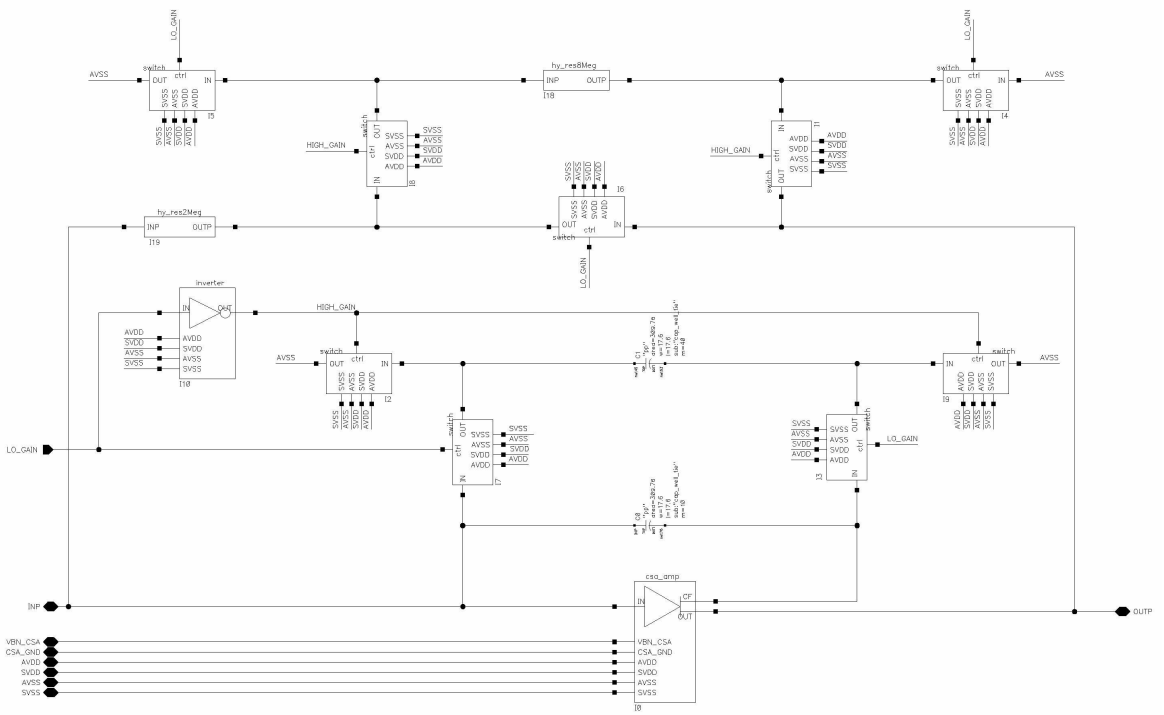
Schematic of the AND gate



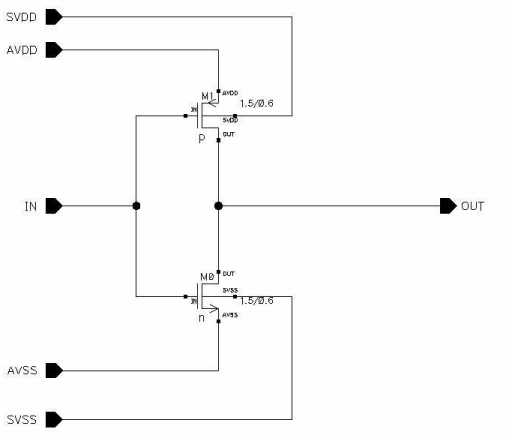
Schematic of the switch used in the channel



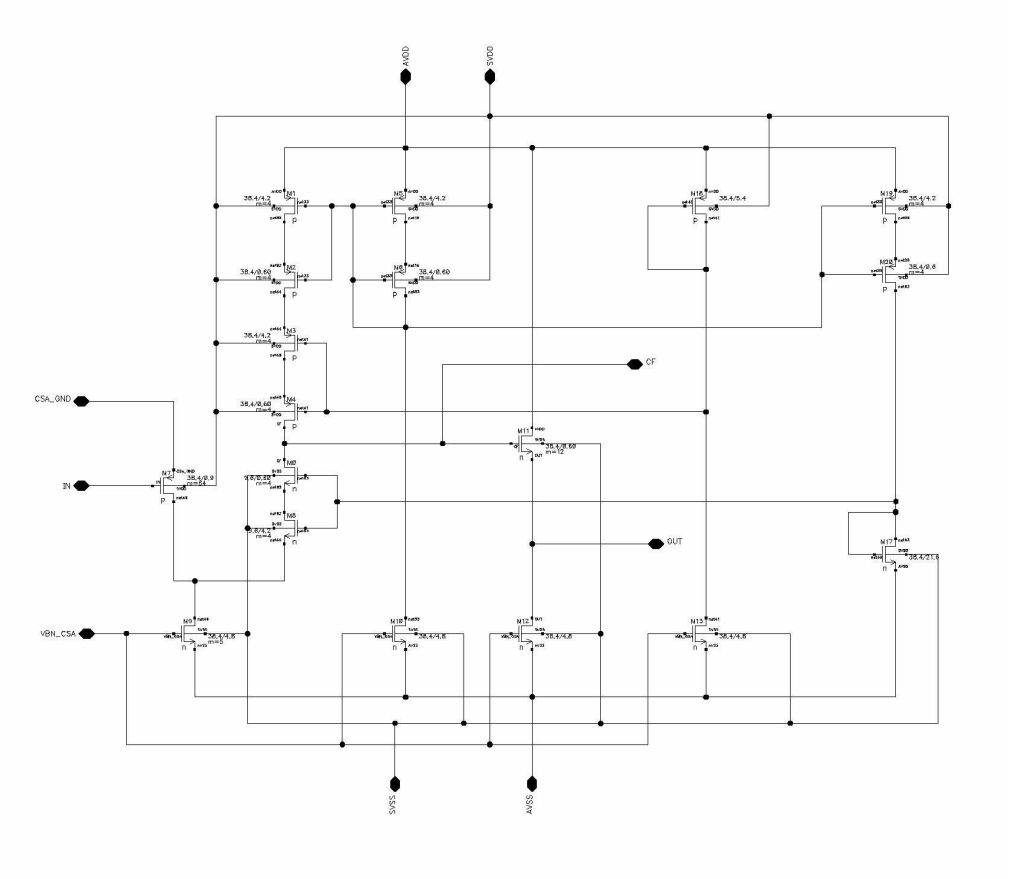
Block diagram of the CSA



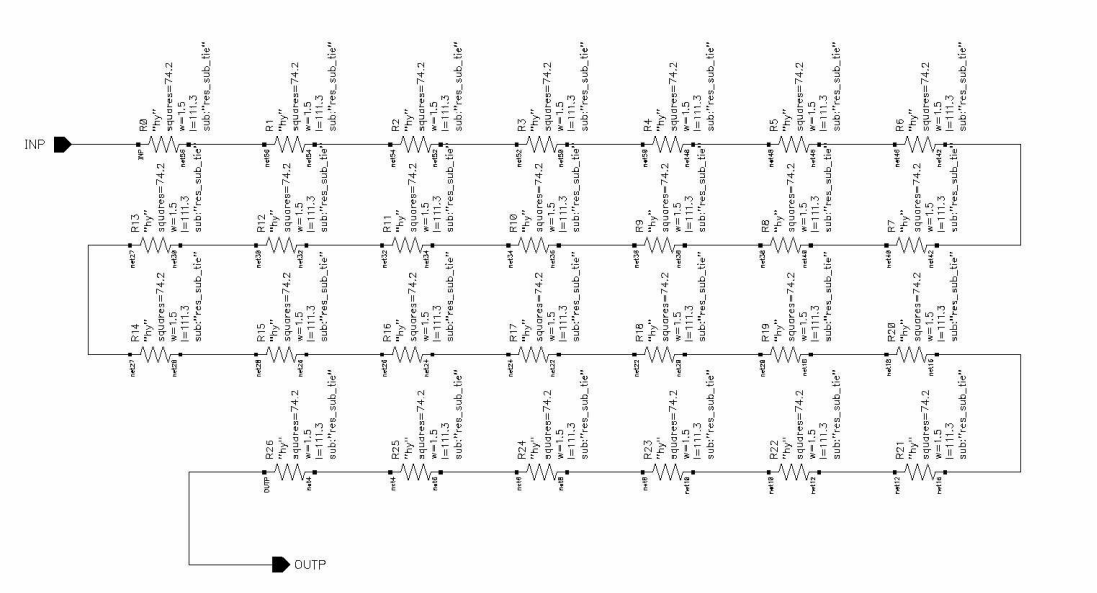
Schematic of the inverter used in CSA



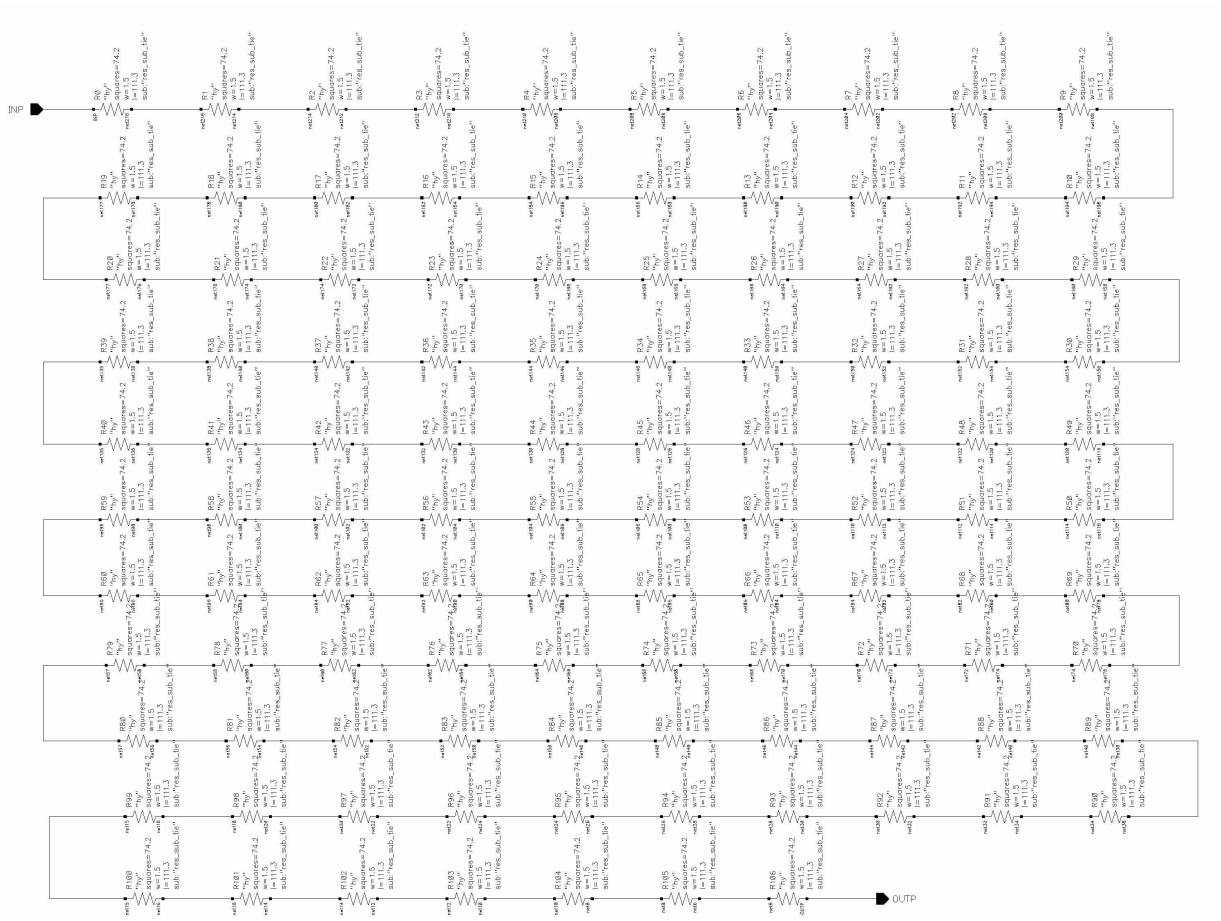
Schematic of the preamplifier used in the CSA



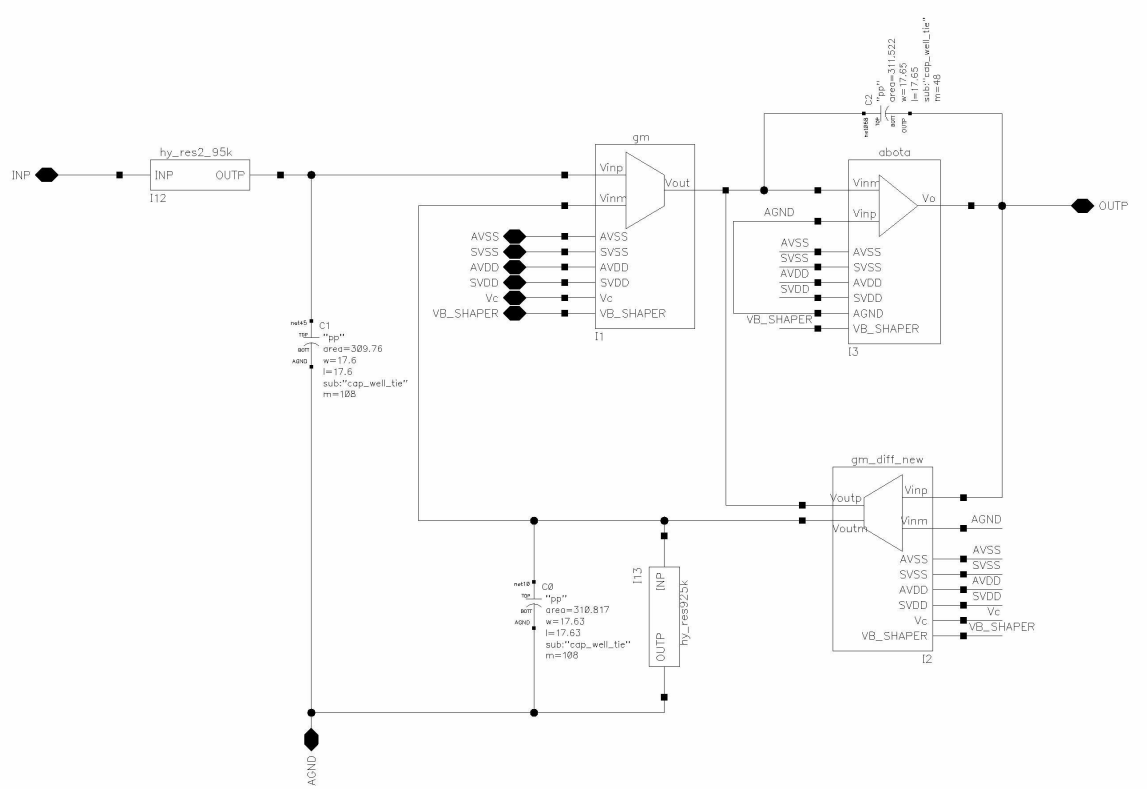
Schematic of the resistor (2Meg) used in the CSA



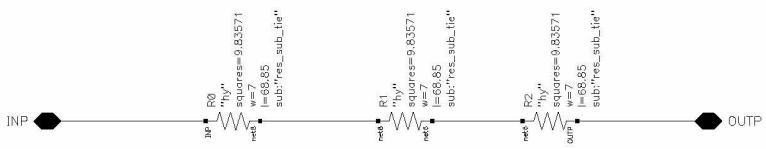
Schematic of the resistor (8Meg) used in the CSA



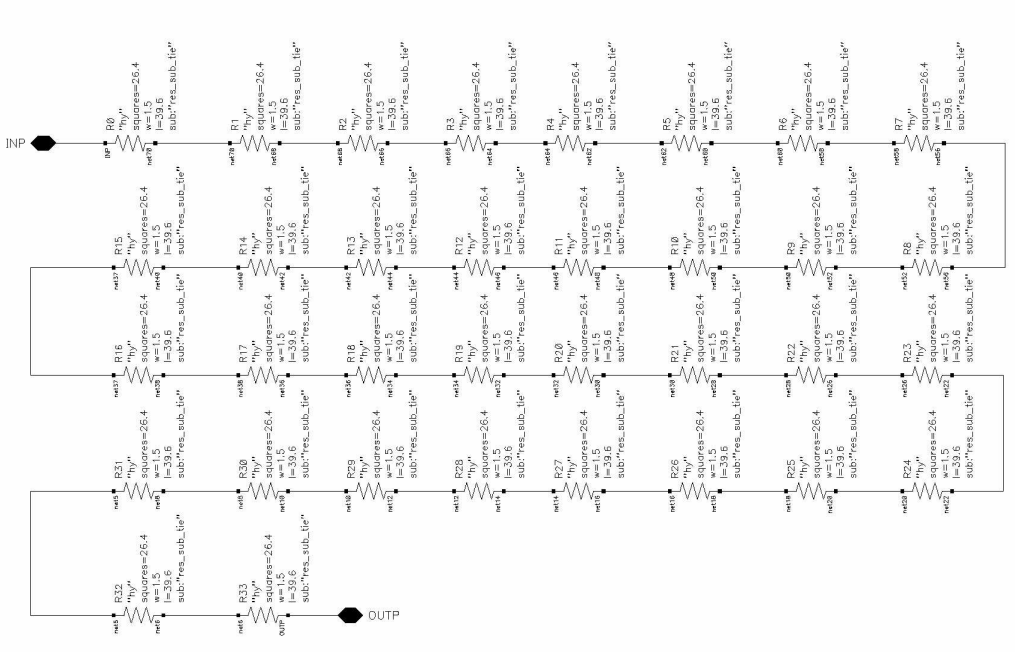
Block diagram of the shaper



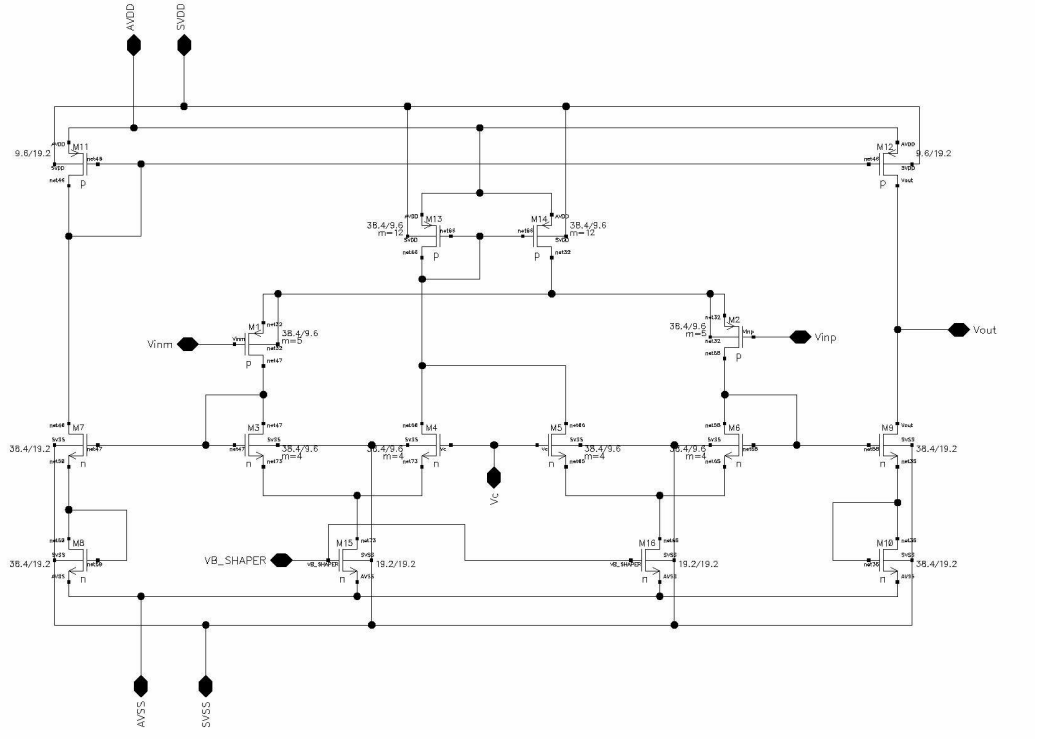
Schematic of the resistor (2.95k) used in the shaper



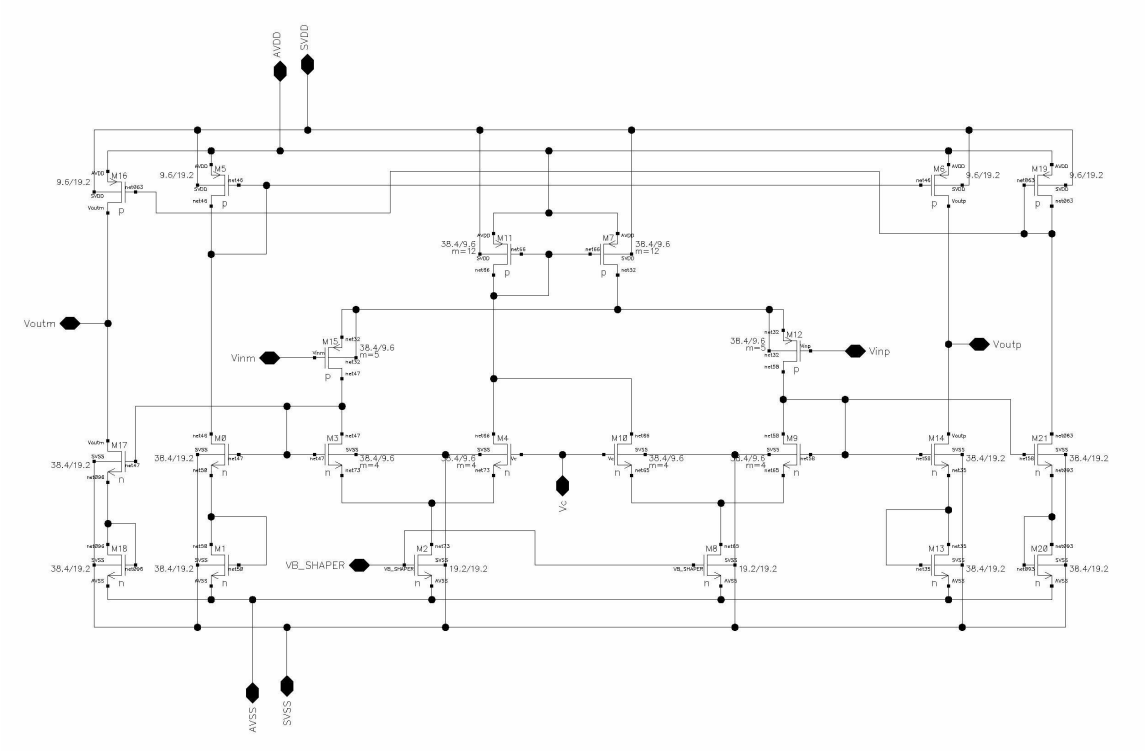
Schematic of the resistor (925k) used in the shaper



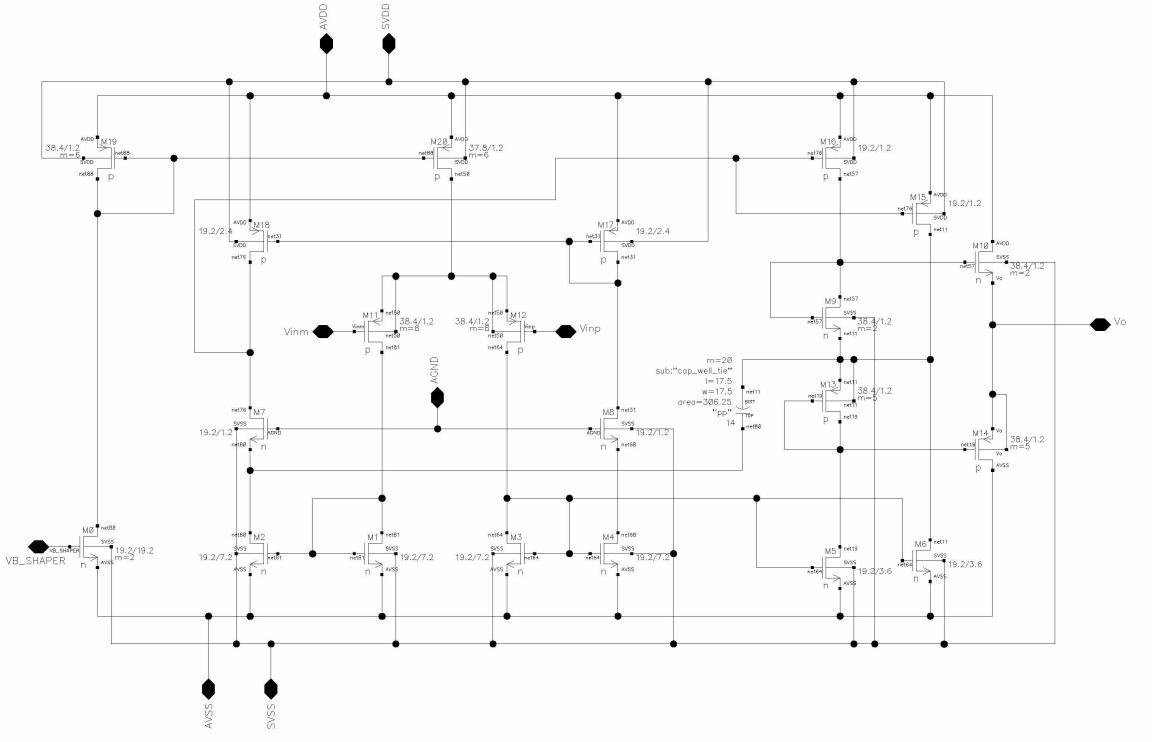
Schematic of the single ended linear transconductor used in the shaper



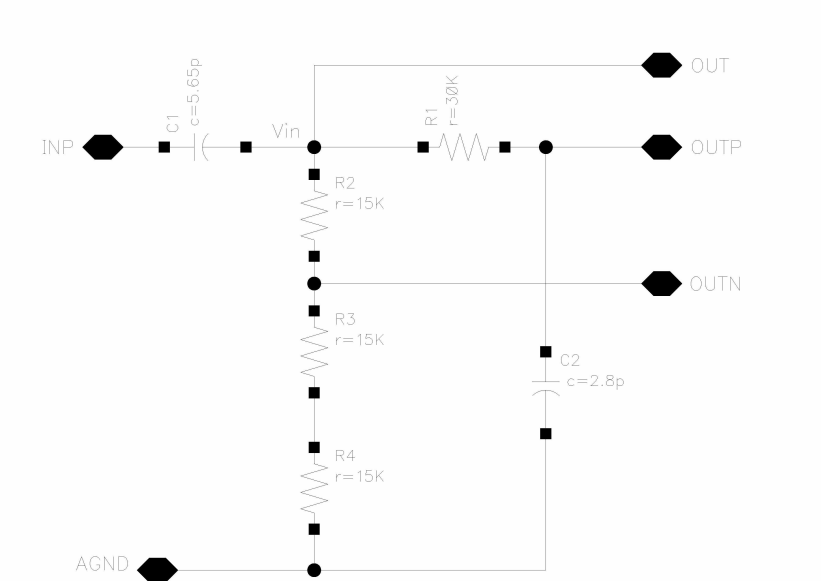
Schematic of the double ended linear transconductor used in the shaper



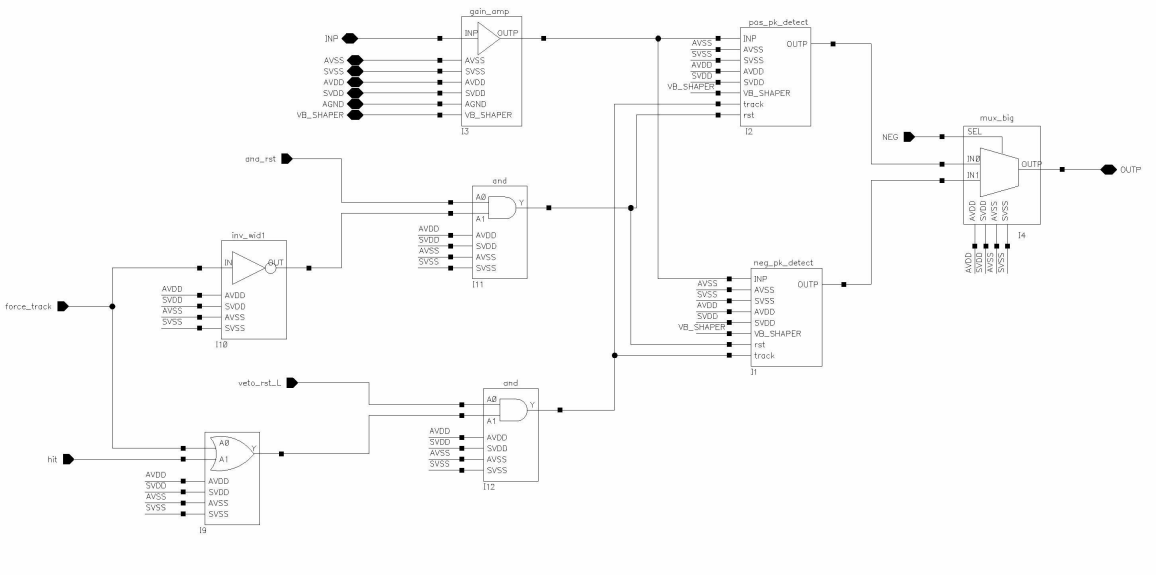
Schematic of the voltage amplifier used in the shaper



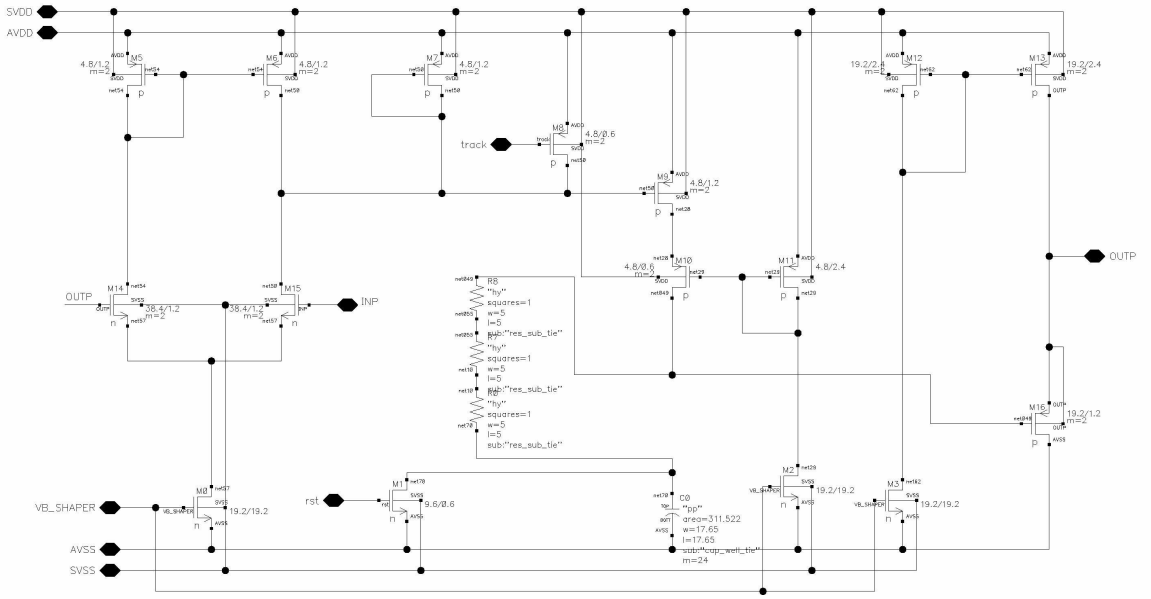
Schematic of the Nowlin circuit



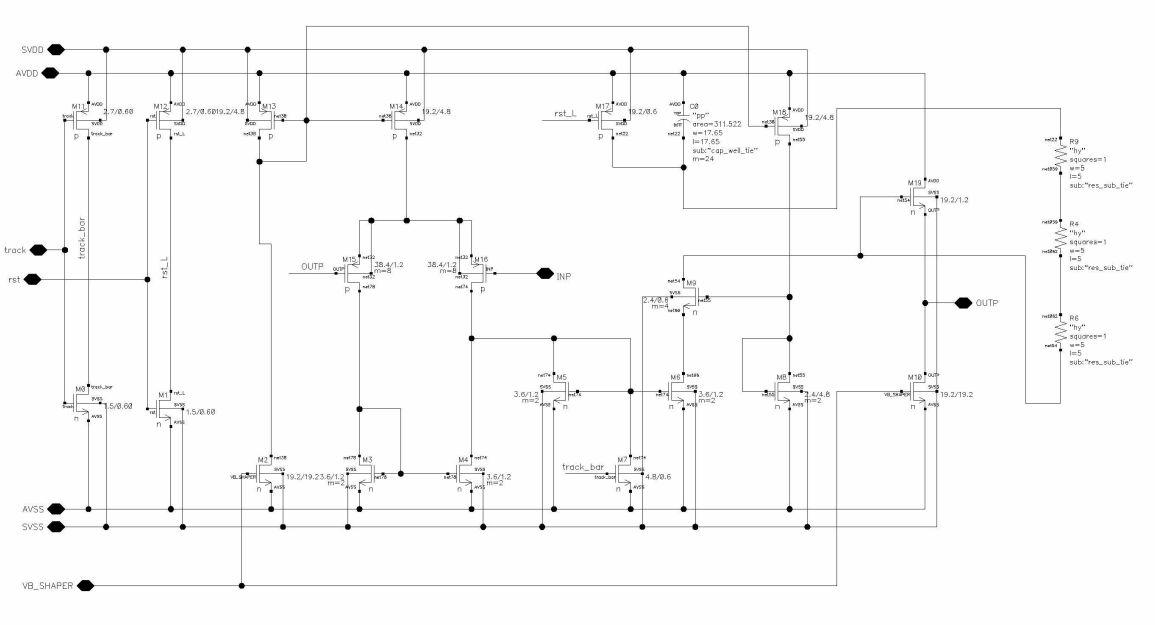
Block diagram of the peak sampler circuit



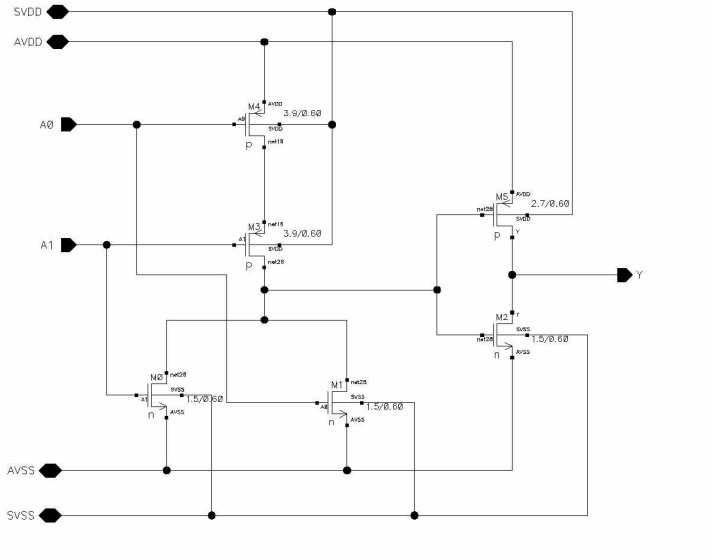
Schematic of the positive peak detector used in the peak sampler



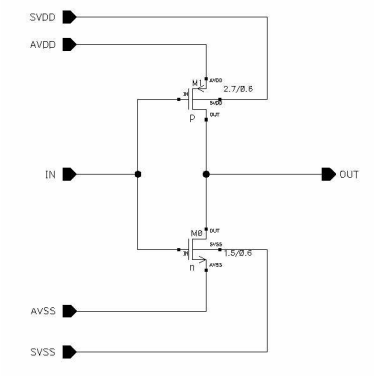
Schematic of the negative peak detector used in the peak sampler



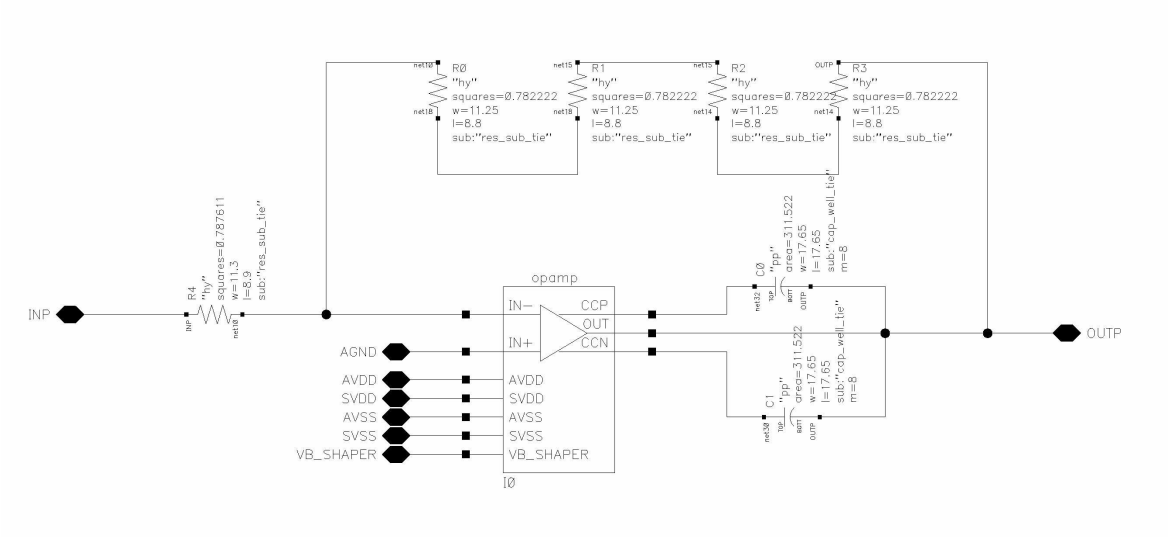
Schematic of the OR gate



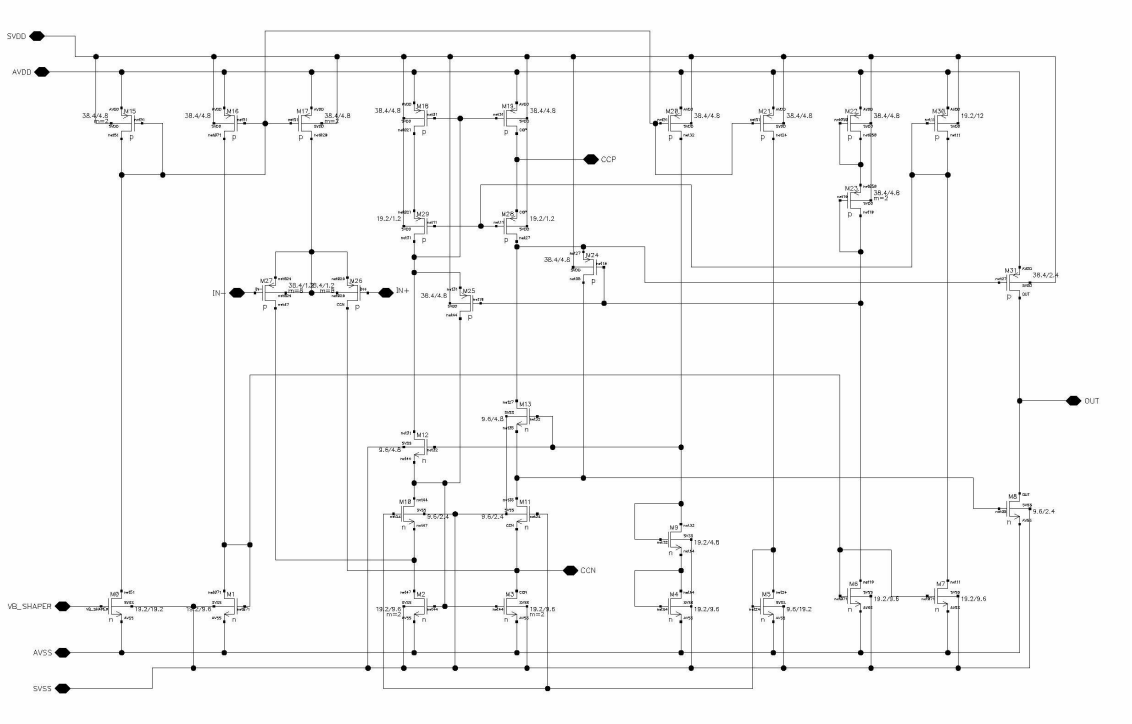
Schematic of the inverter



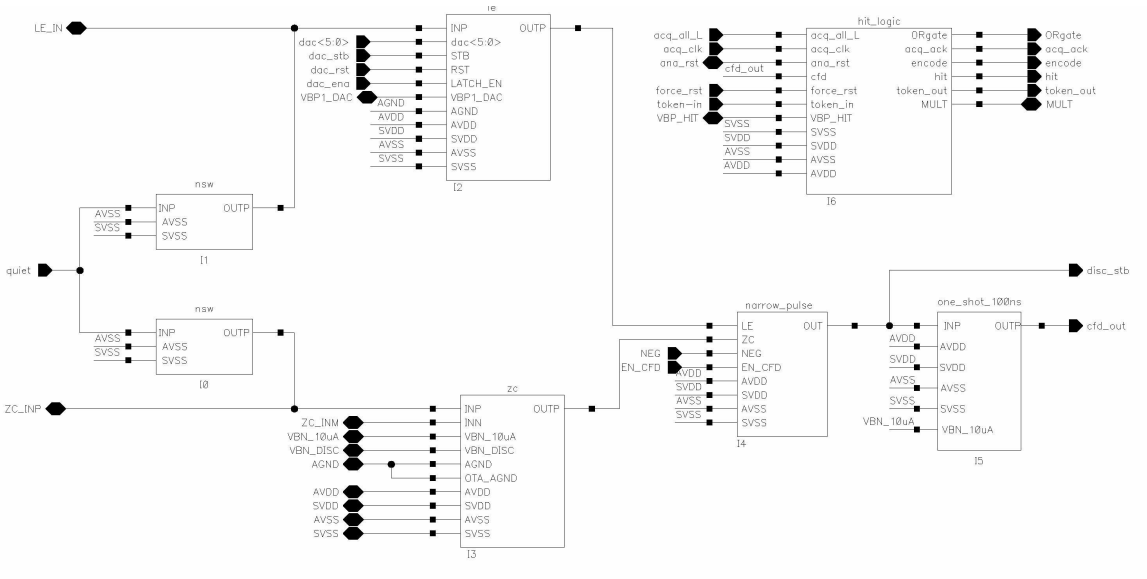
Schematic of the gain amplifier used in the peak sampler



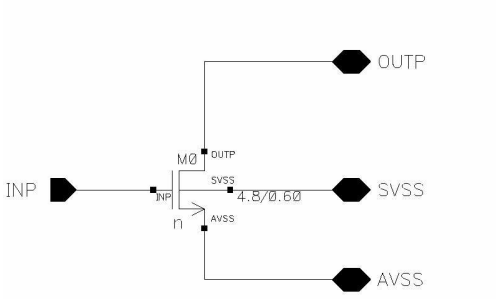
Schematic of the opamp used in the gain amplifier



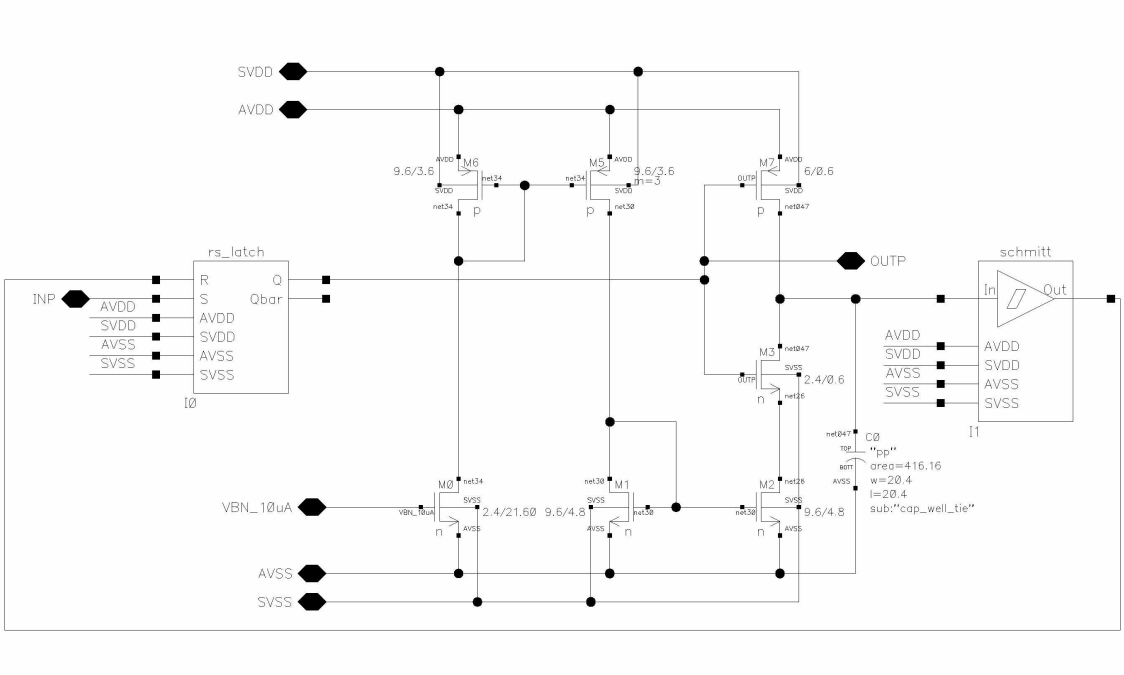
Block diagram of the constant fraction discriminator



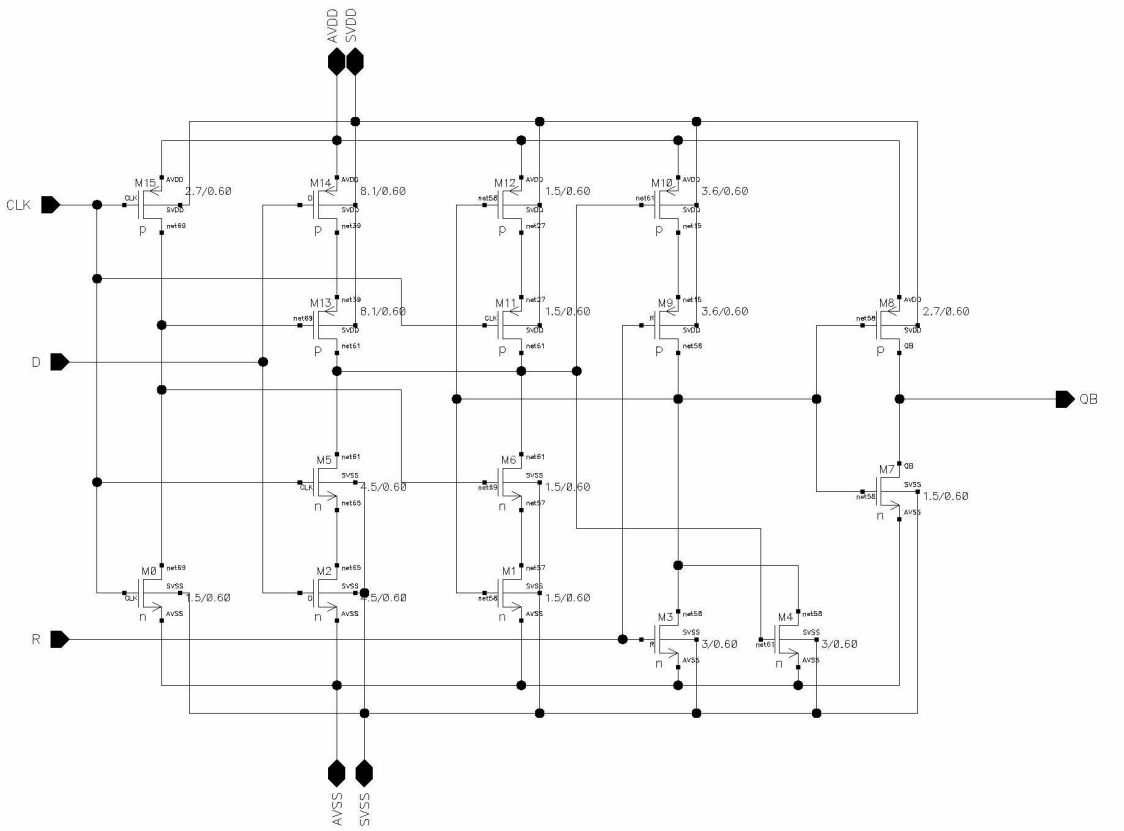
Schematic of the N-Type switch used in CFD



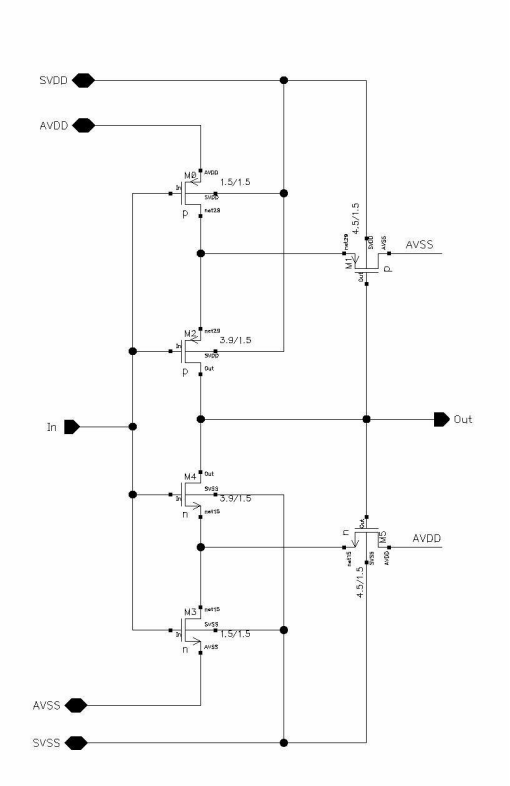
Schematic of the one shot used in the CFD



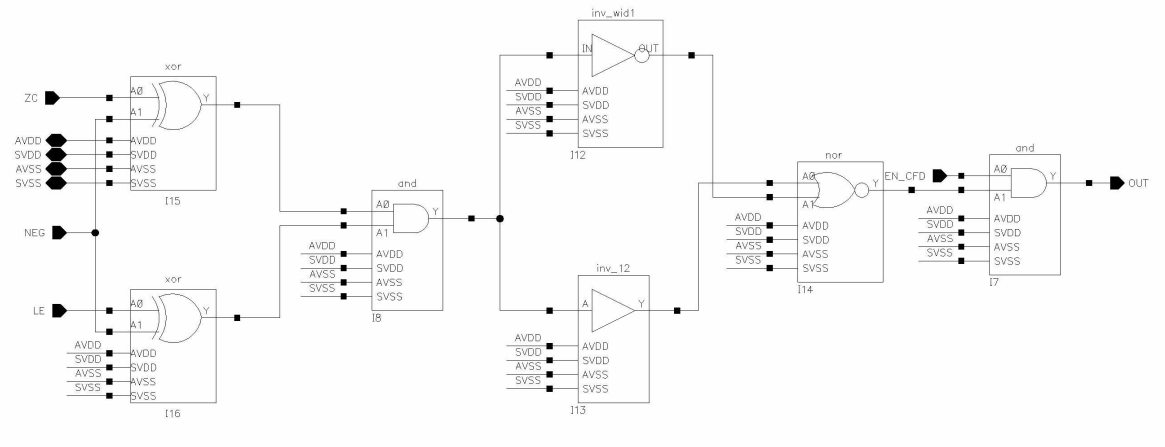
Schematic of the latch used in the one shot



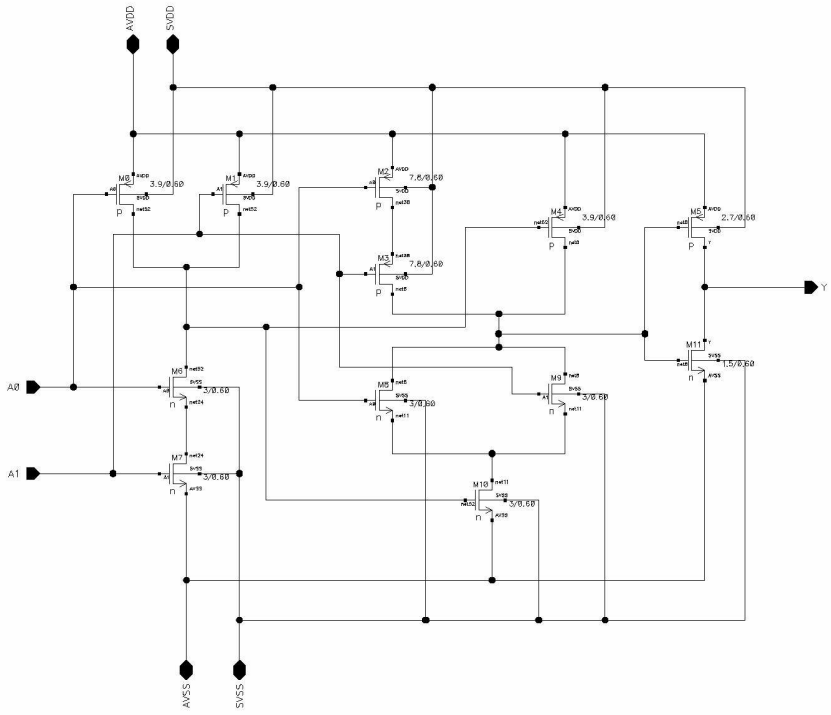
Schematic of the Schmitt trigger used in the one shot



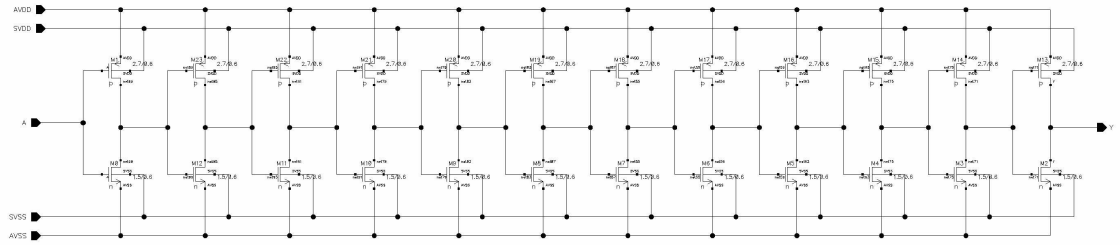
Block diagram of the narrow pulse circuit used in the CFD



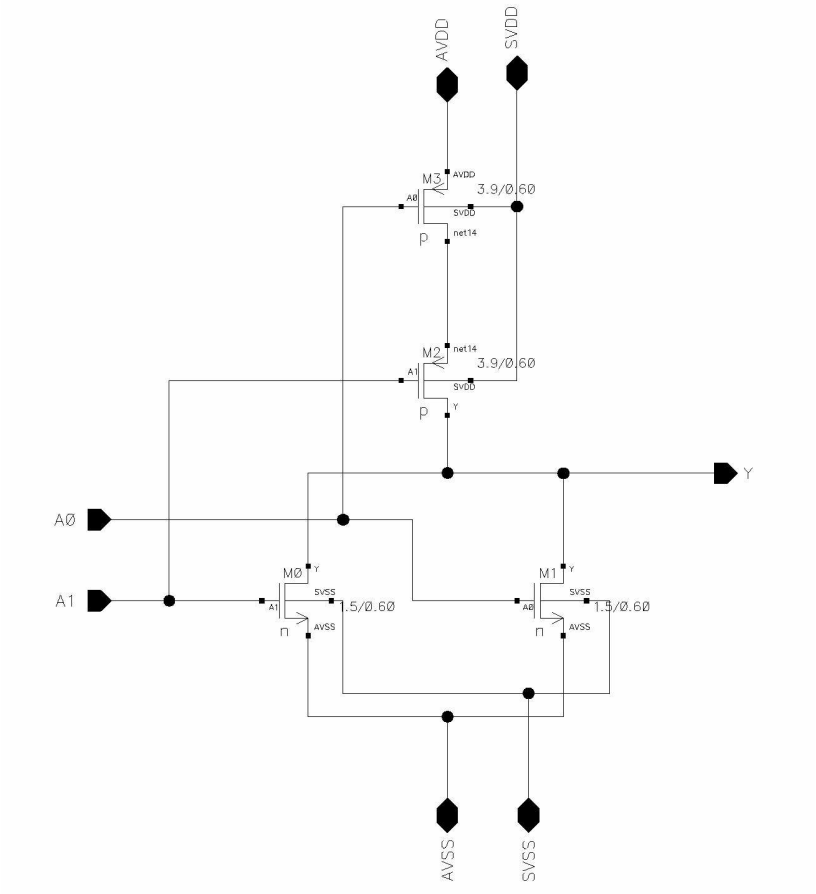
Schematic of the XOR gate



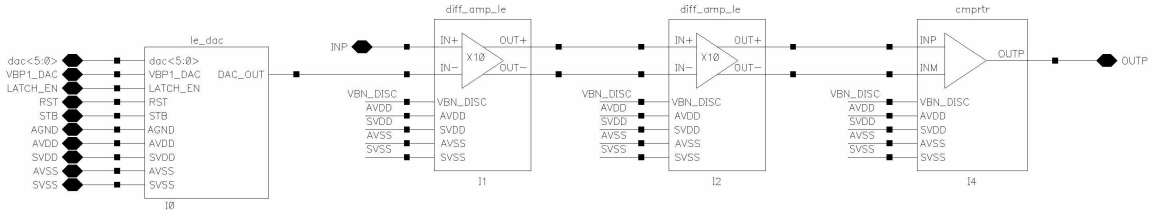
Schematic of the delay circuit (inv_12) used in the narrow pulse



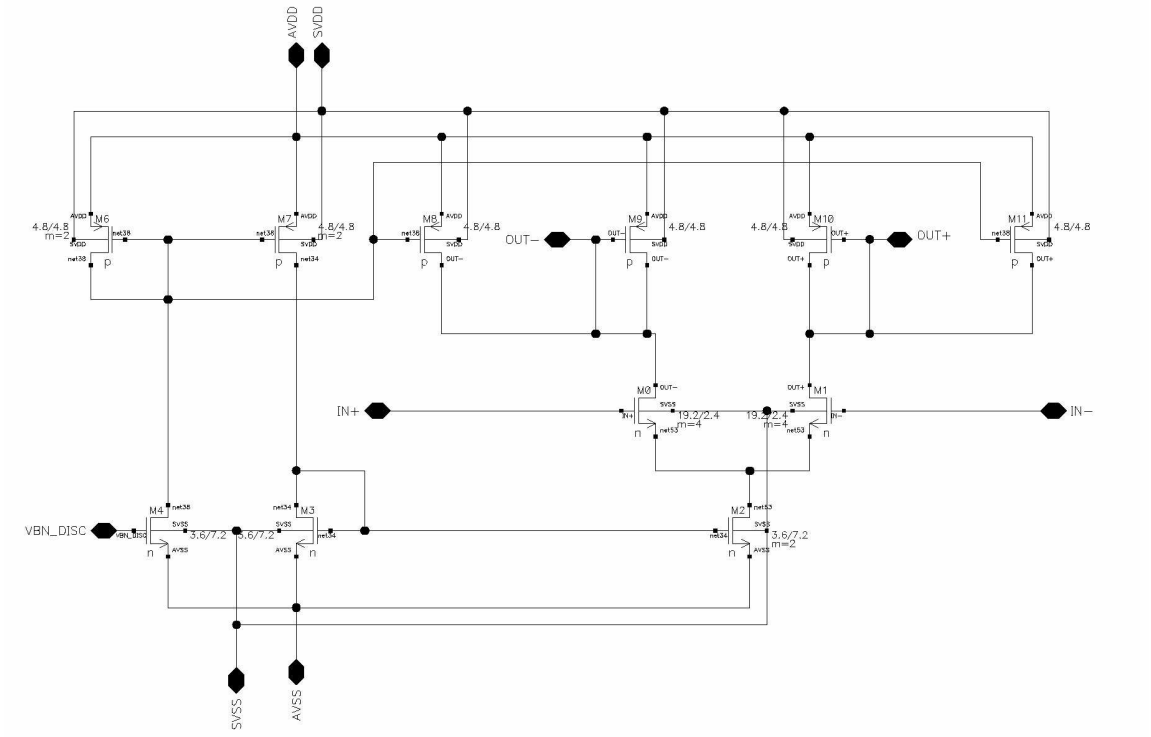
Schematic of the NOR gate



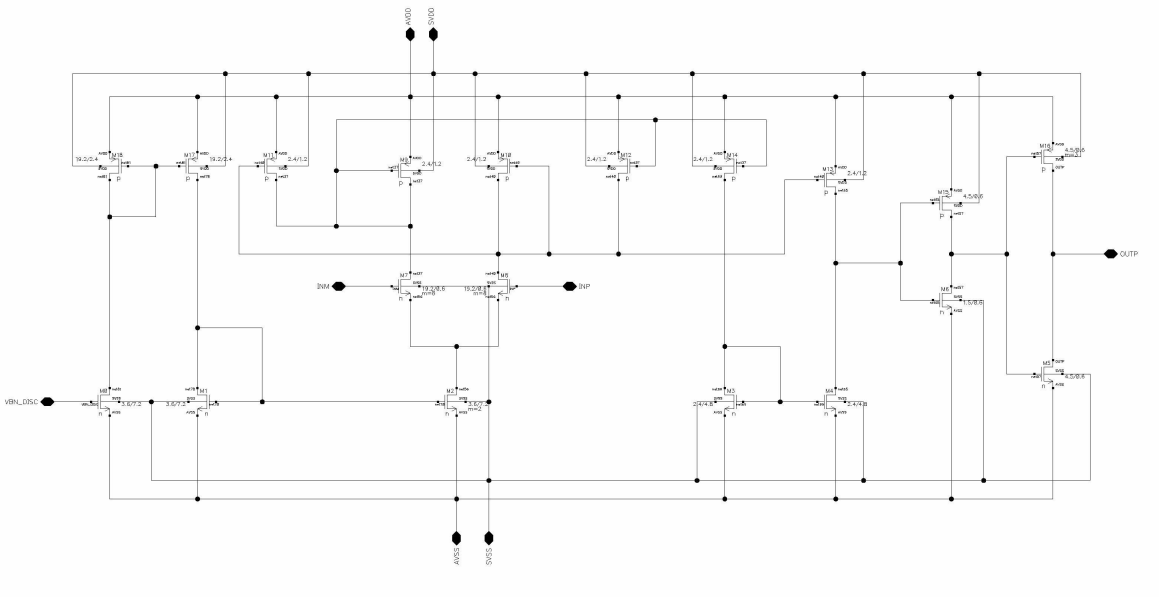
Block diagram of the leading edge discriminator



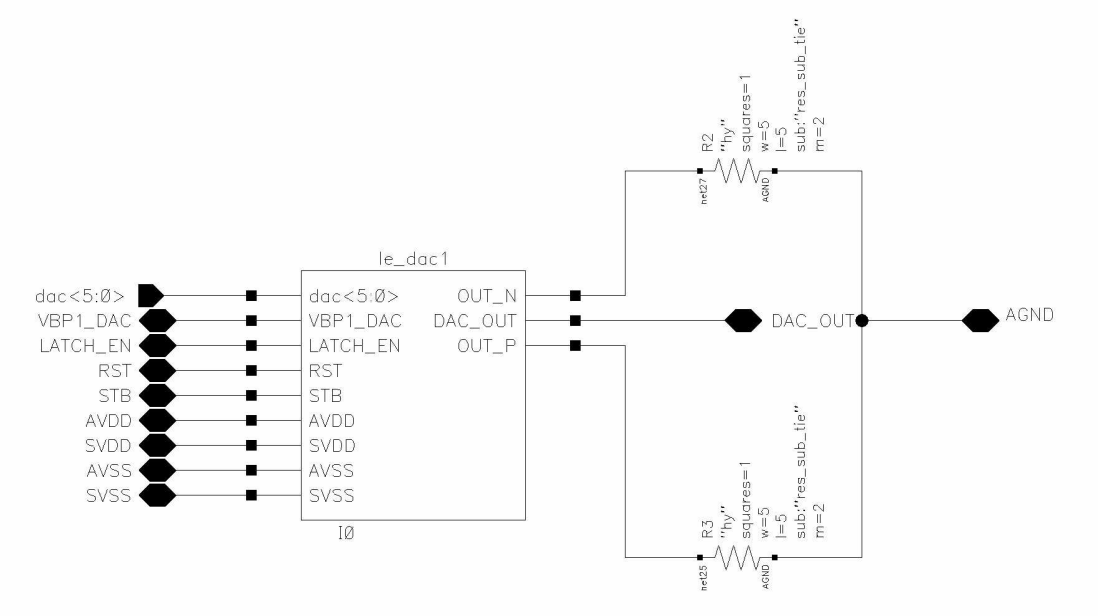
Schematic of the differential amplifier used in LE



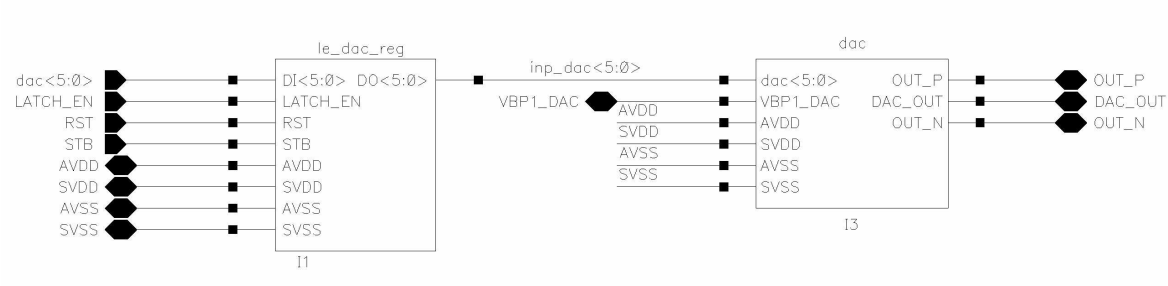
Schematic of the comparator used in LE and ZC



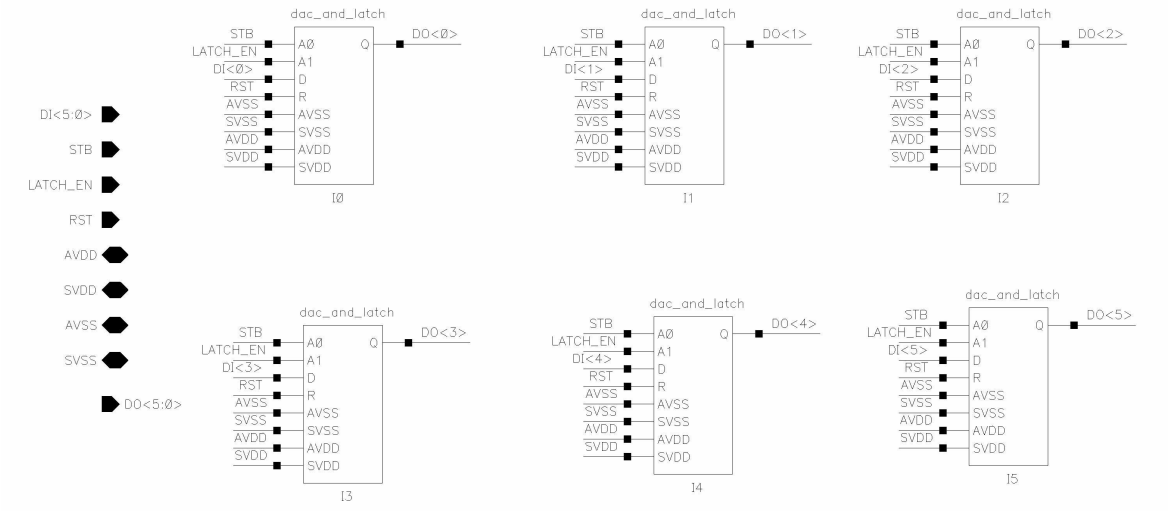
Block diagram of DAC with resistors used in LE



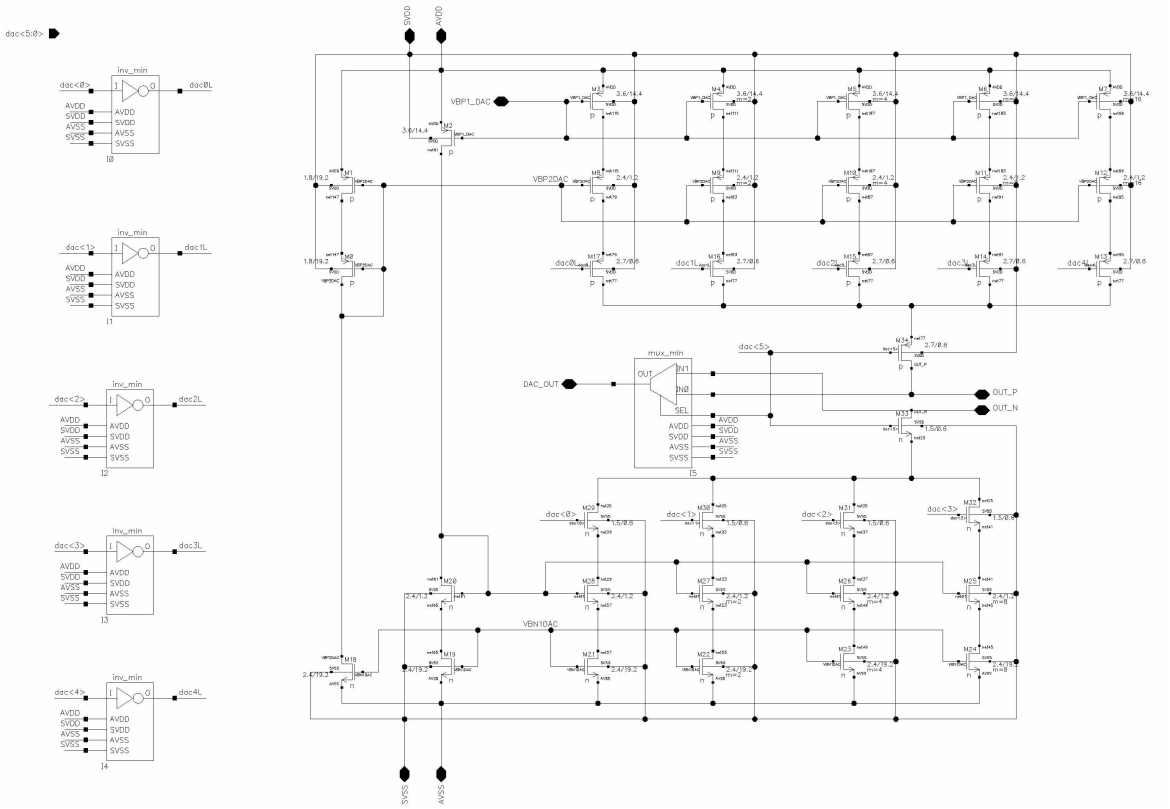
Block diagram of the DAC and registers used in LE



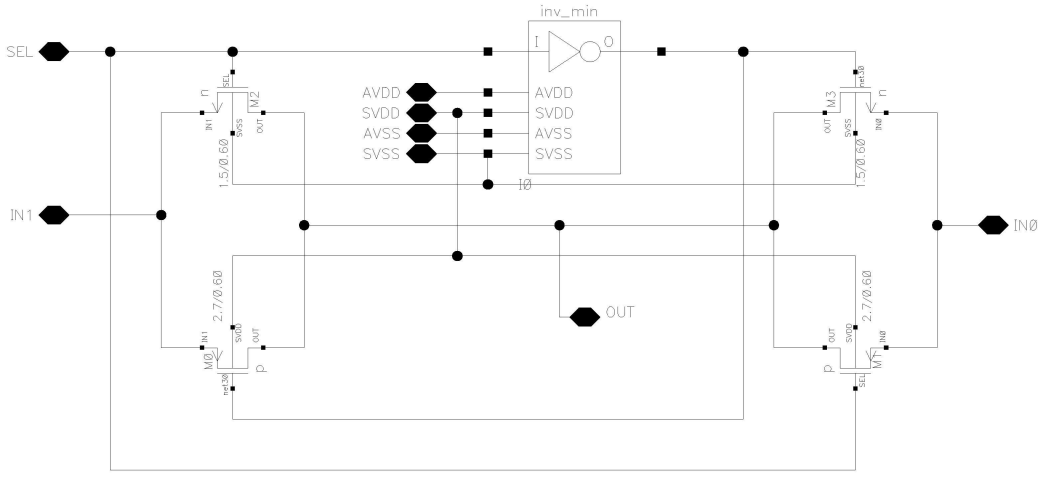
Schematic of the registers used in the DAC



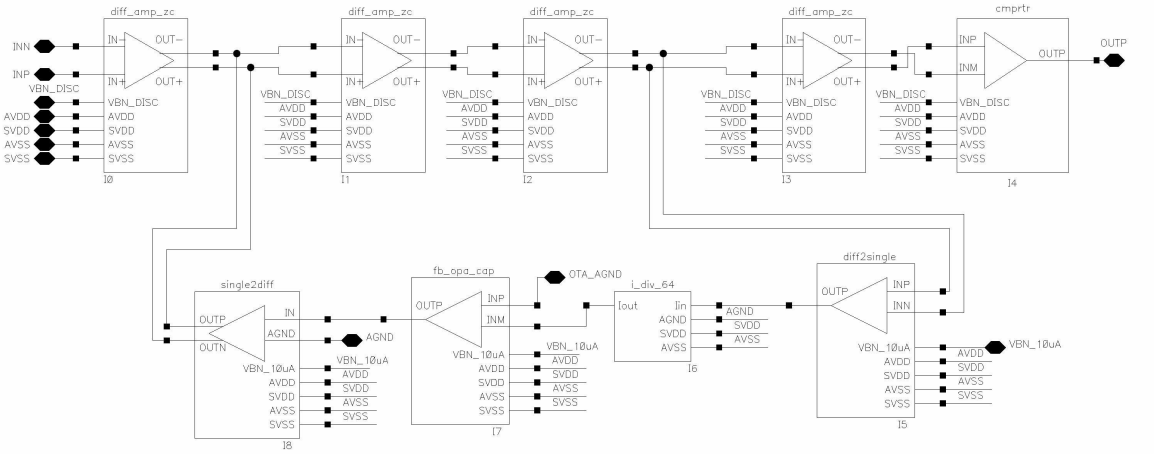
Schematic of the DAC used in LE



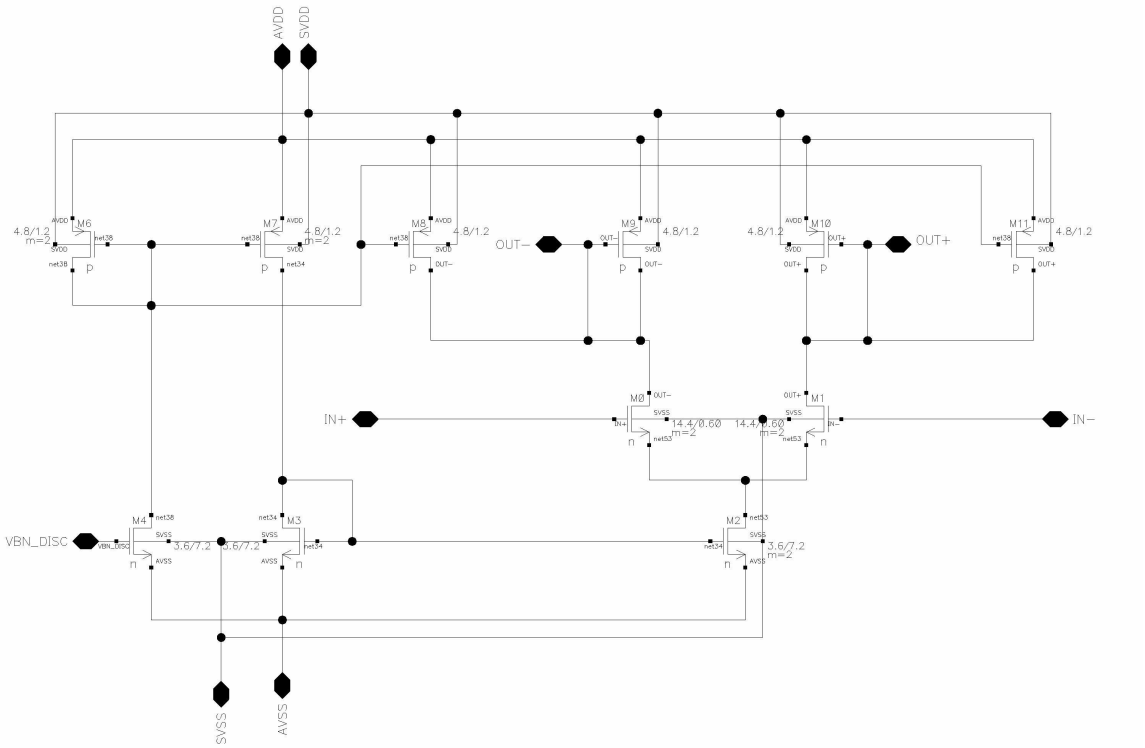
Schematic of the multiplexer used in DAC



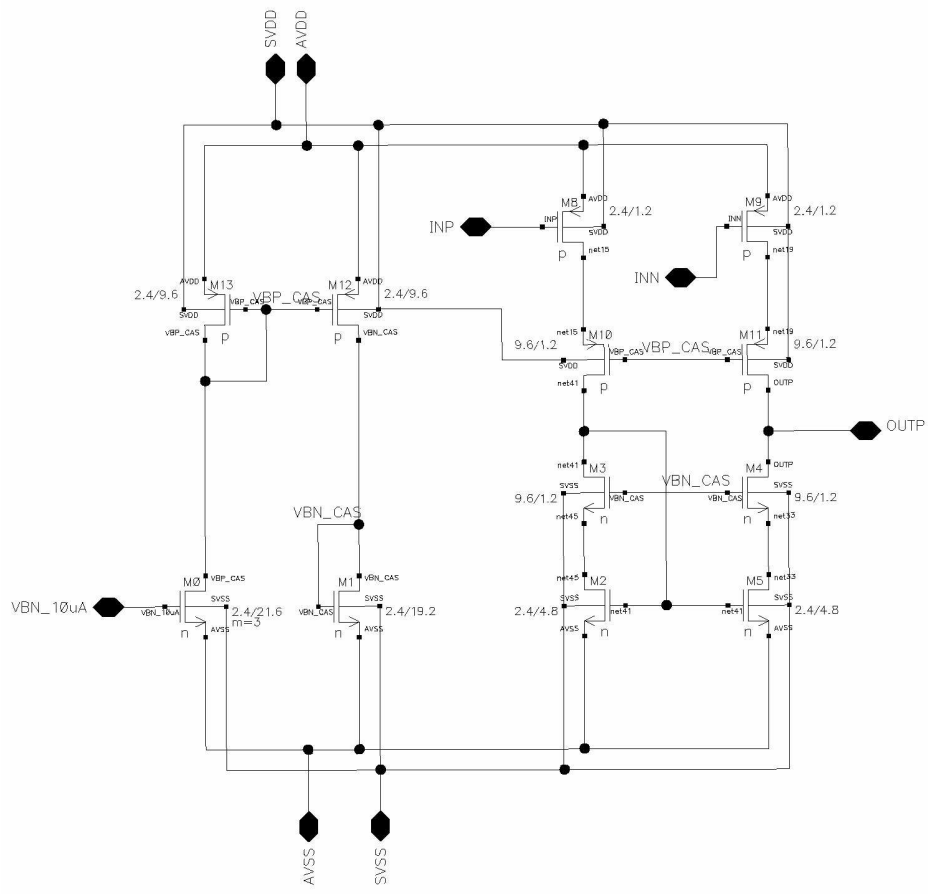
Block diagram of the zero crossing discriminator



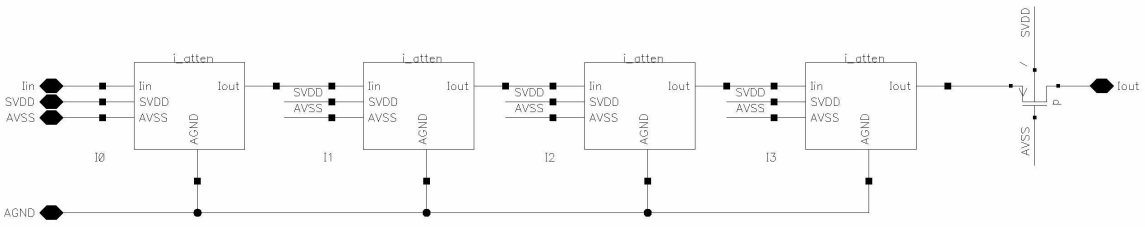
Schematic of the differential amplifier used in ZC



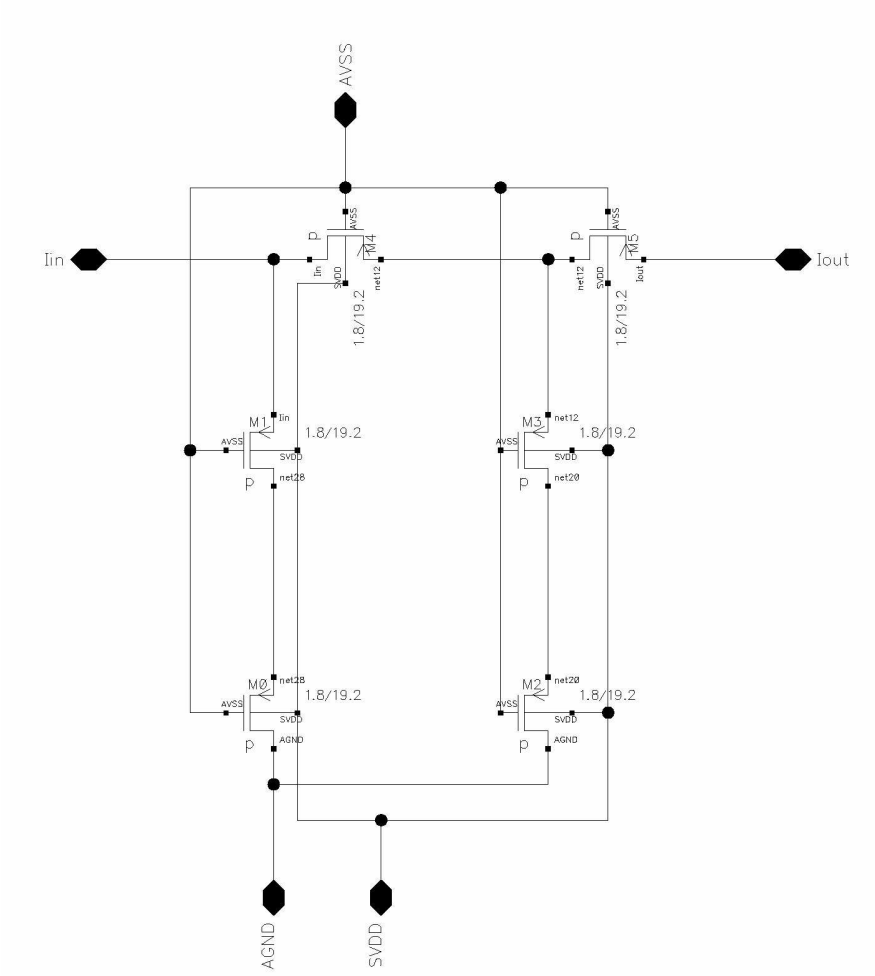
Schematic of the diff2single circuit used in ZC



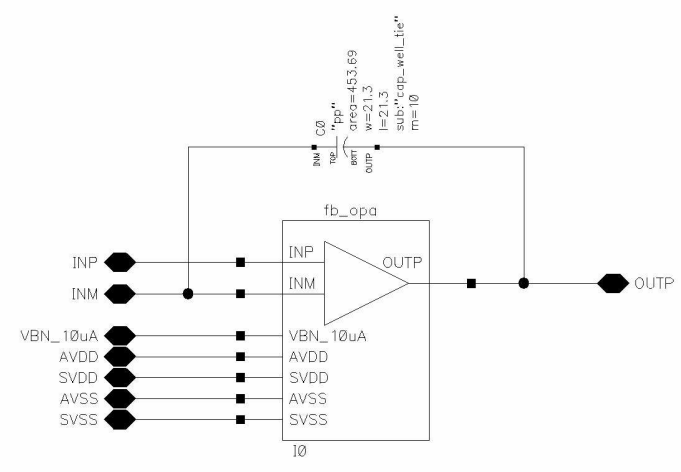
Block diagram of the current divider used in ZC



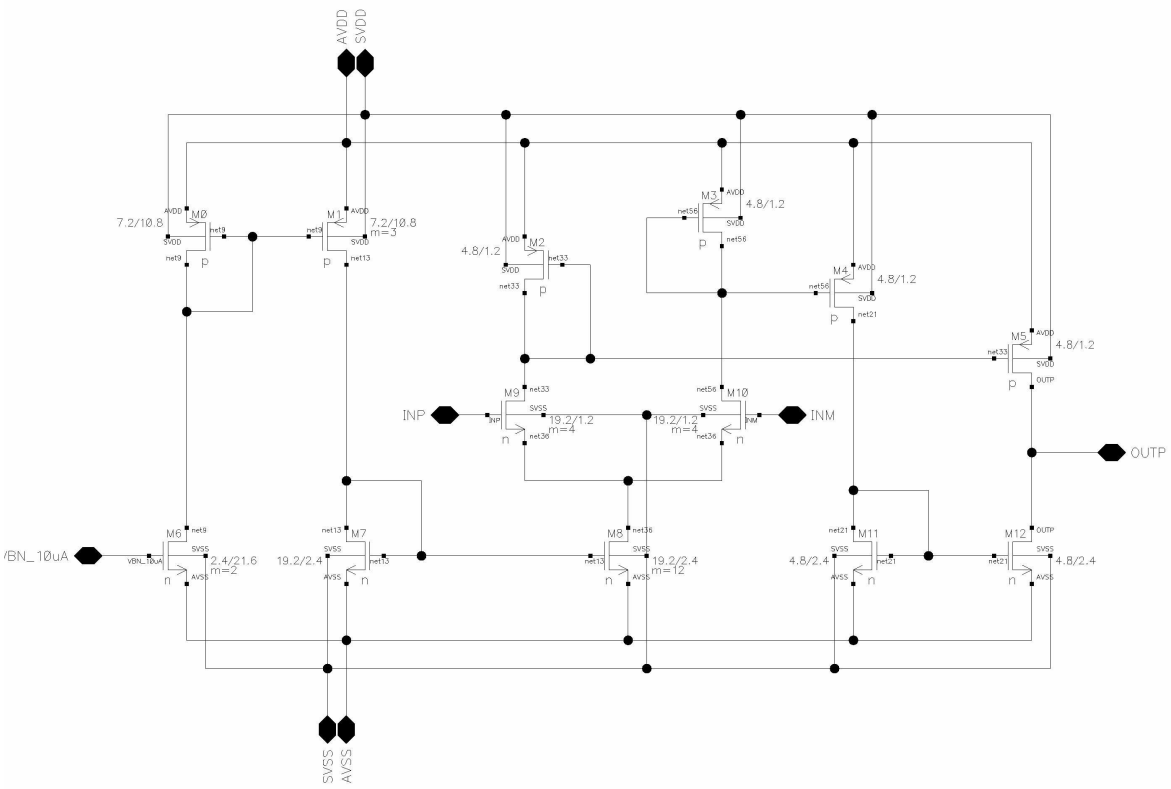
Schematic of the current attenuator used in the current divider



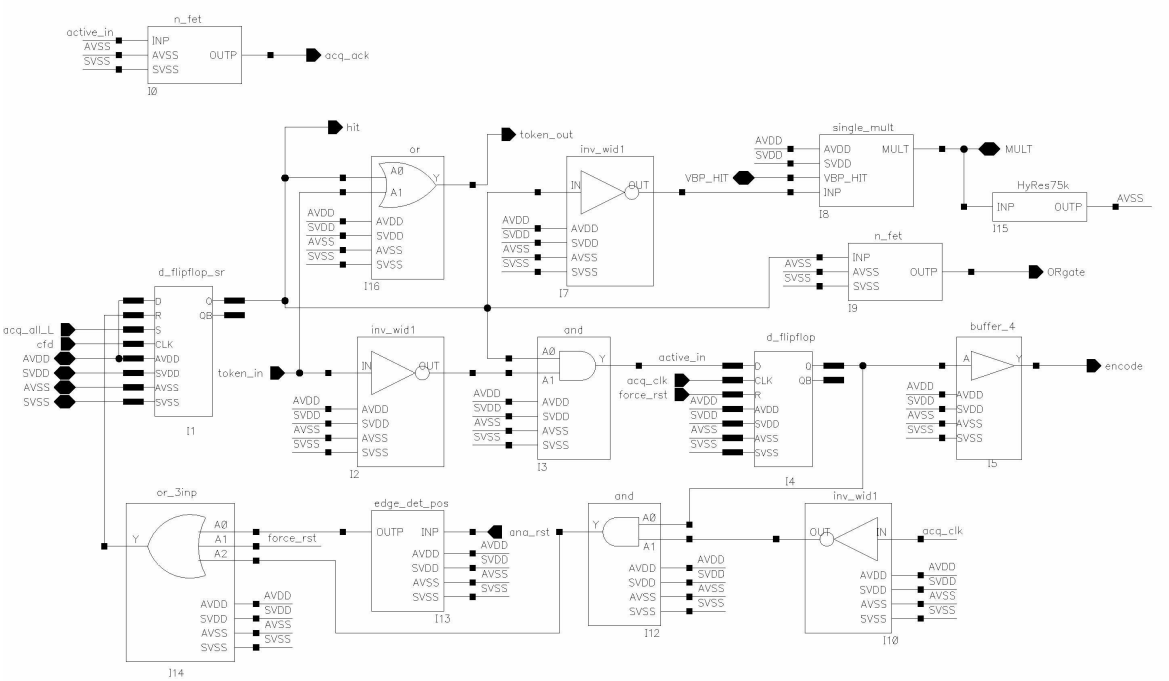
Schematic of the feedback opamp with capacitor feedback used in ZC



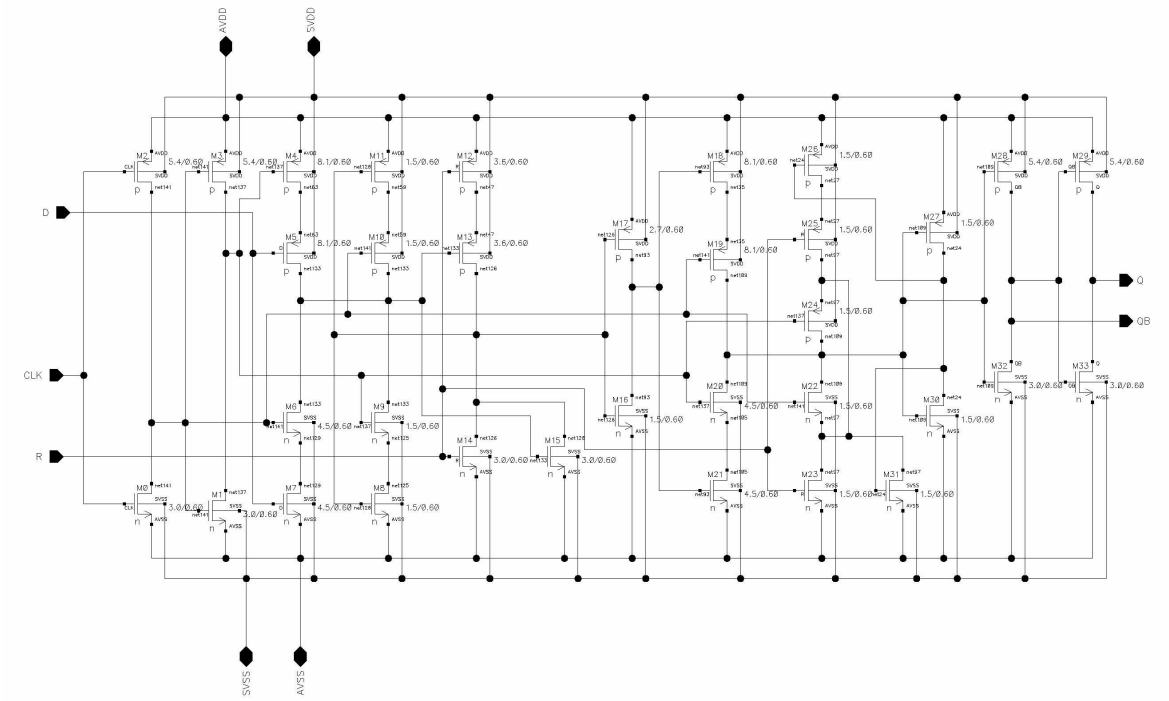
Schematic of the feedback amplifier used in ZC



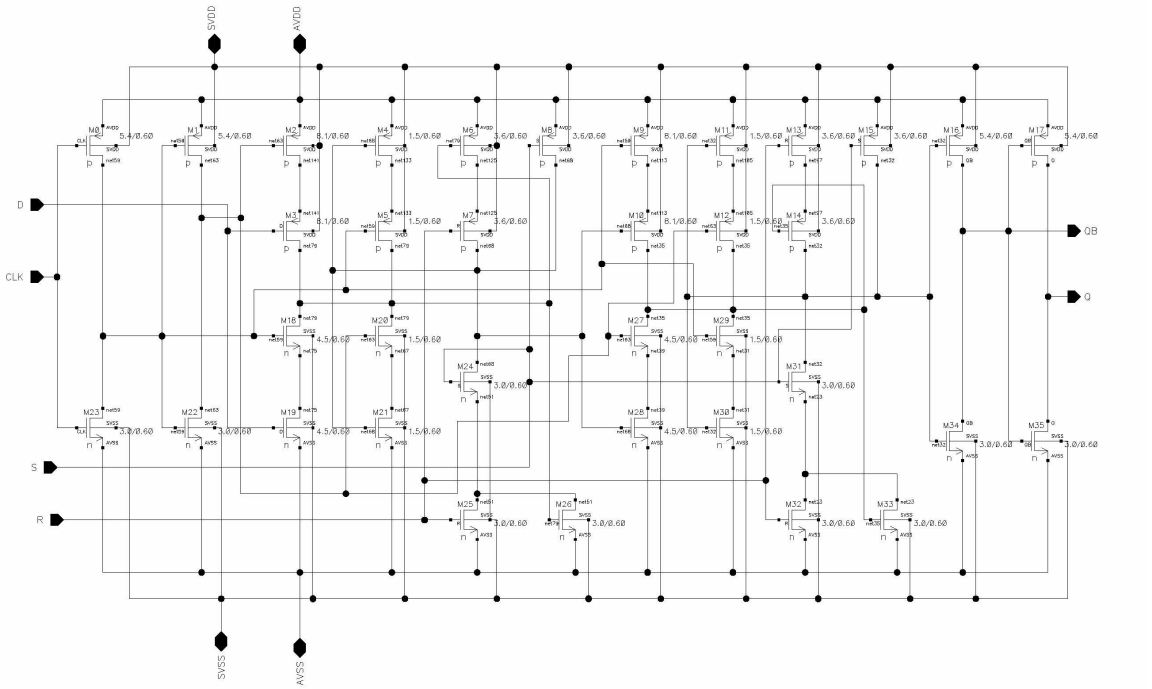
Block diagram of the hit logic



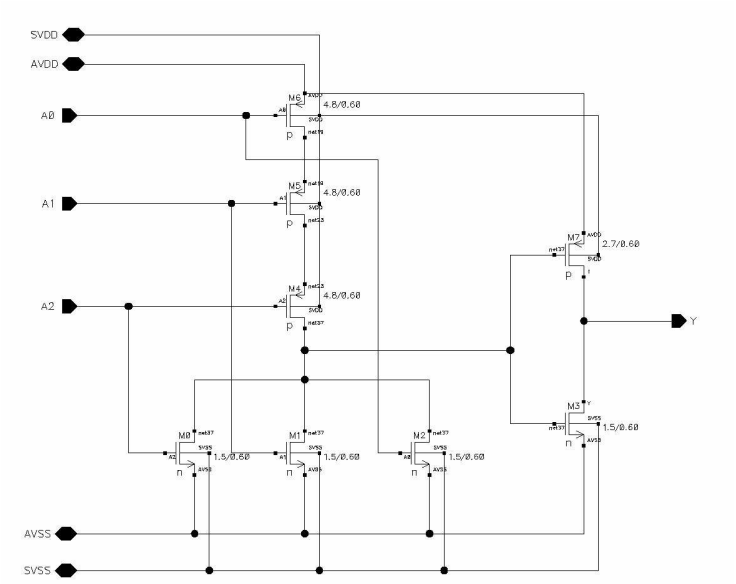
Schematic of the D-flipflop



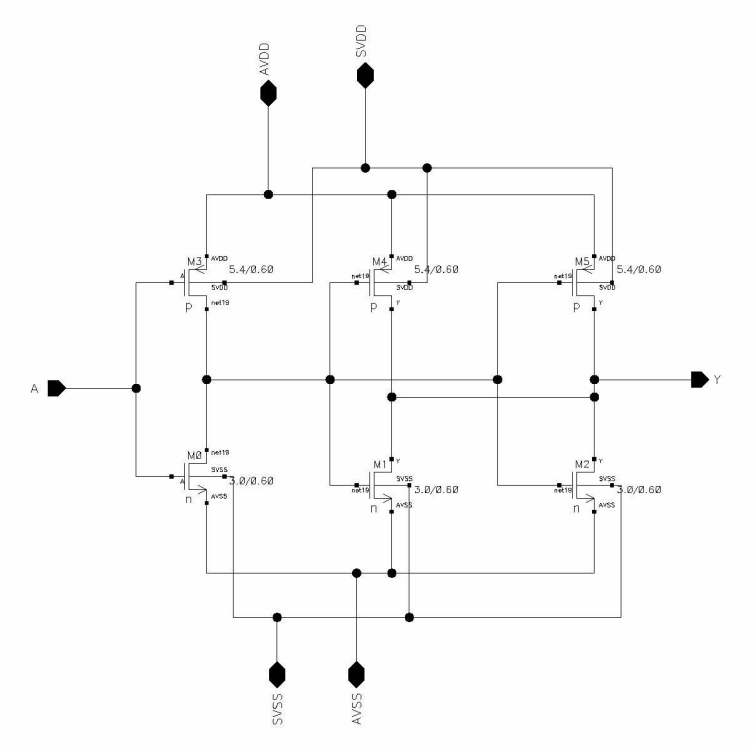
Schematic of the D-flipflop with set and reset



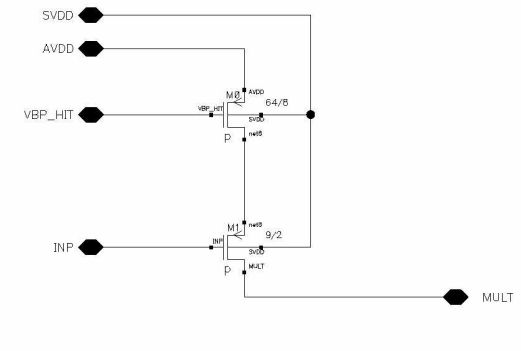
Schematic of the 3 input OR gate



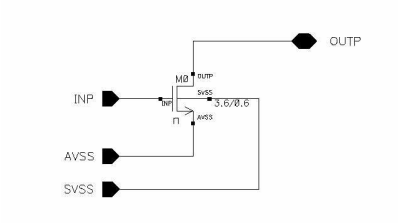
Schematic of the buffer used in the hit logic



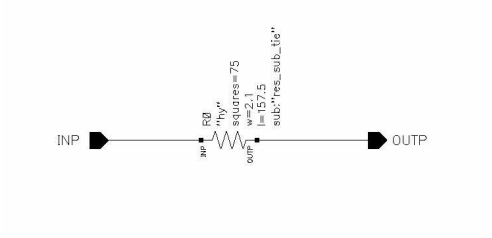
Schematic of the single_mult circuit used in the hit logic



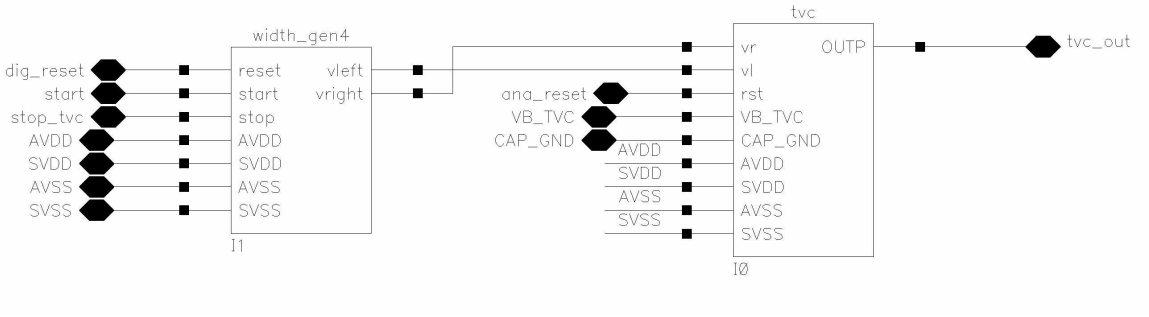
Schematic of the N-fet used in the hit logic



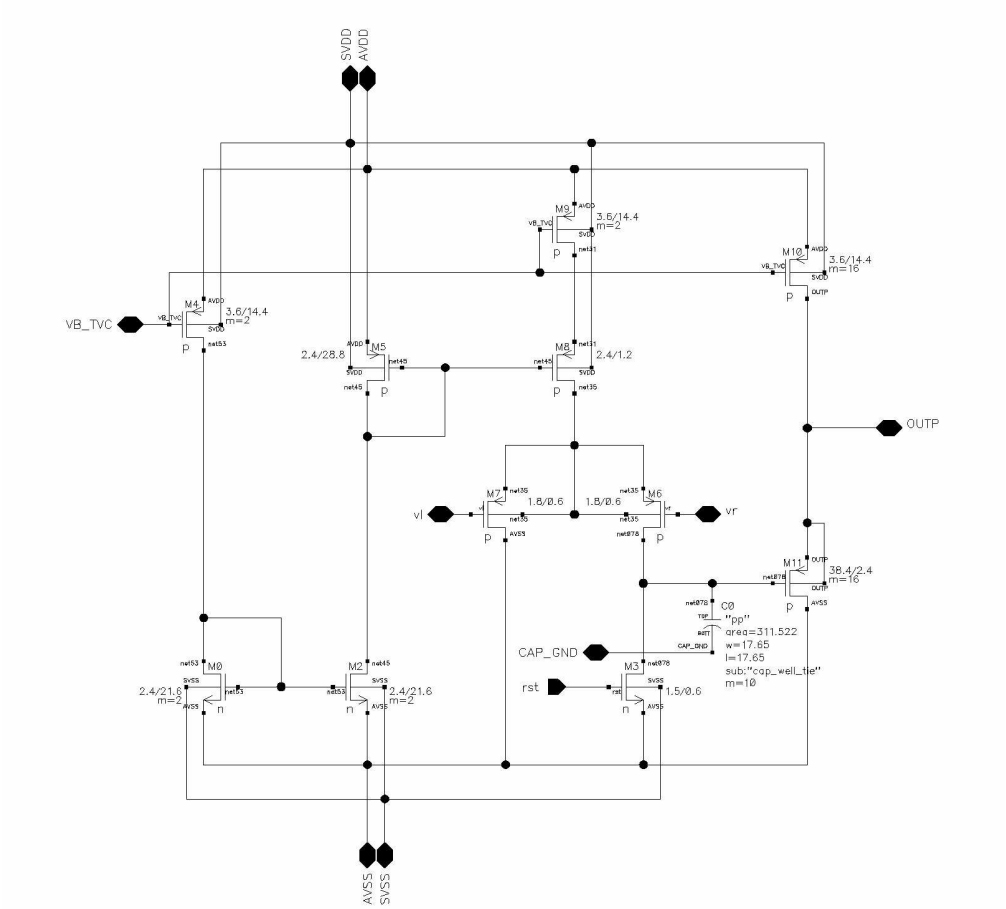
Schematic of the resistor (75k) used in the hit logic



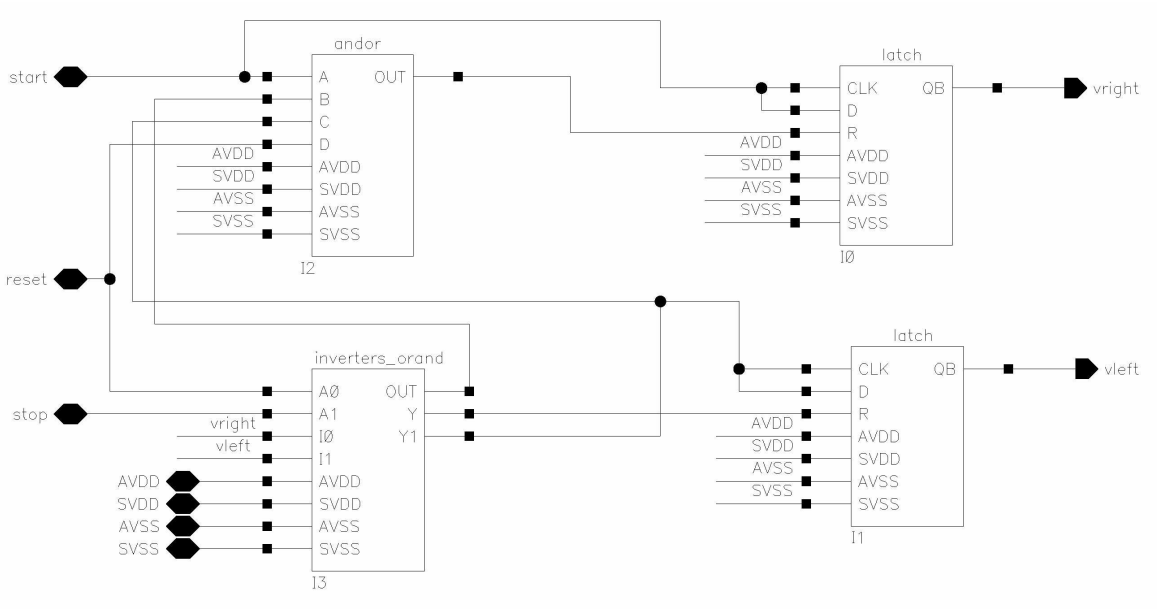
Block diagram of the time time to voltage converter



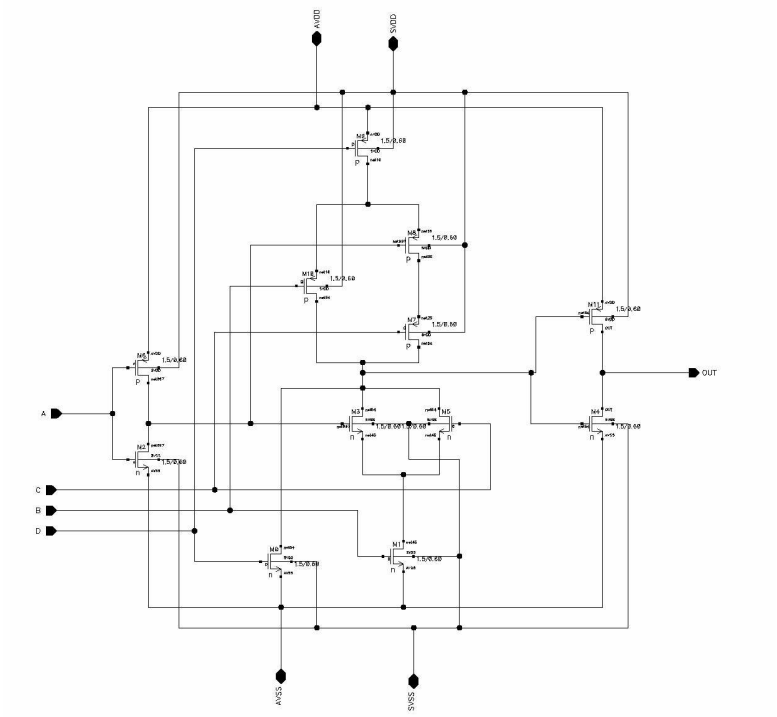
Schematic of the TVC



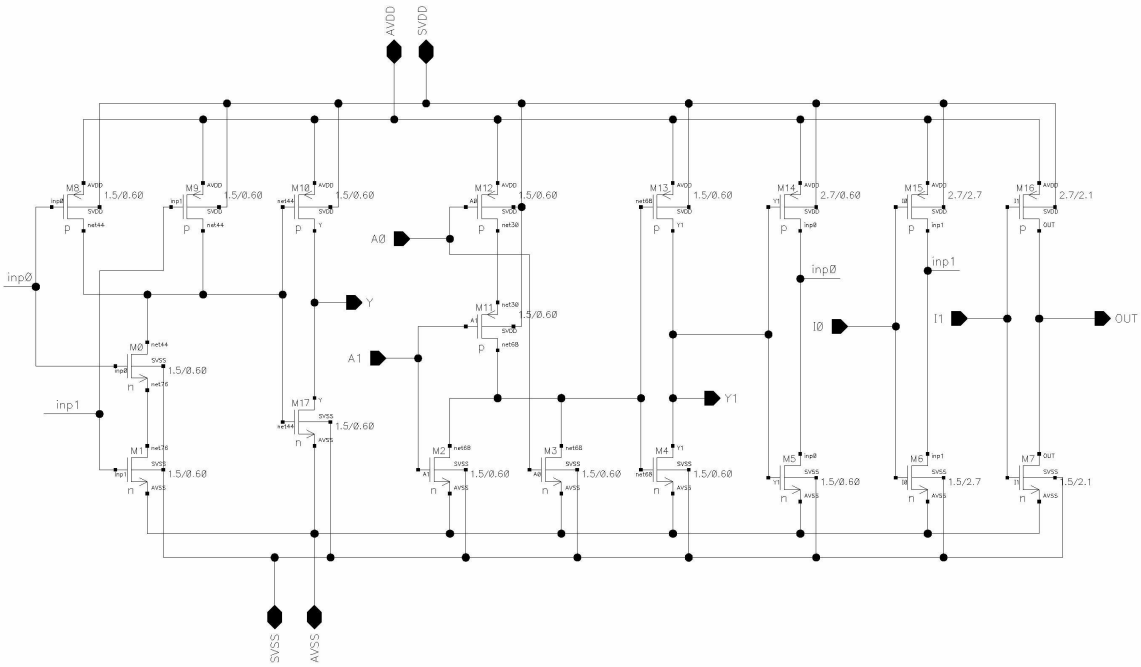
Block diagram of the width generator used in the TVC



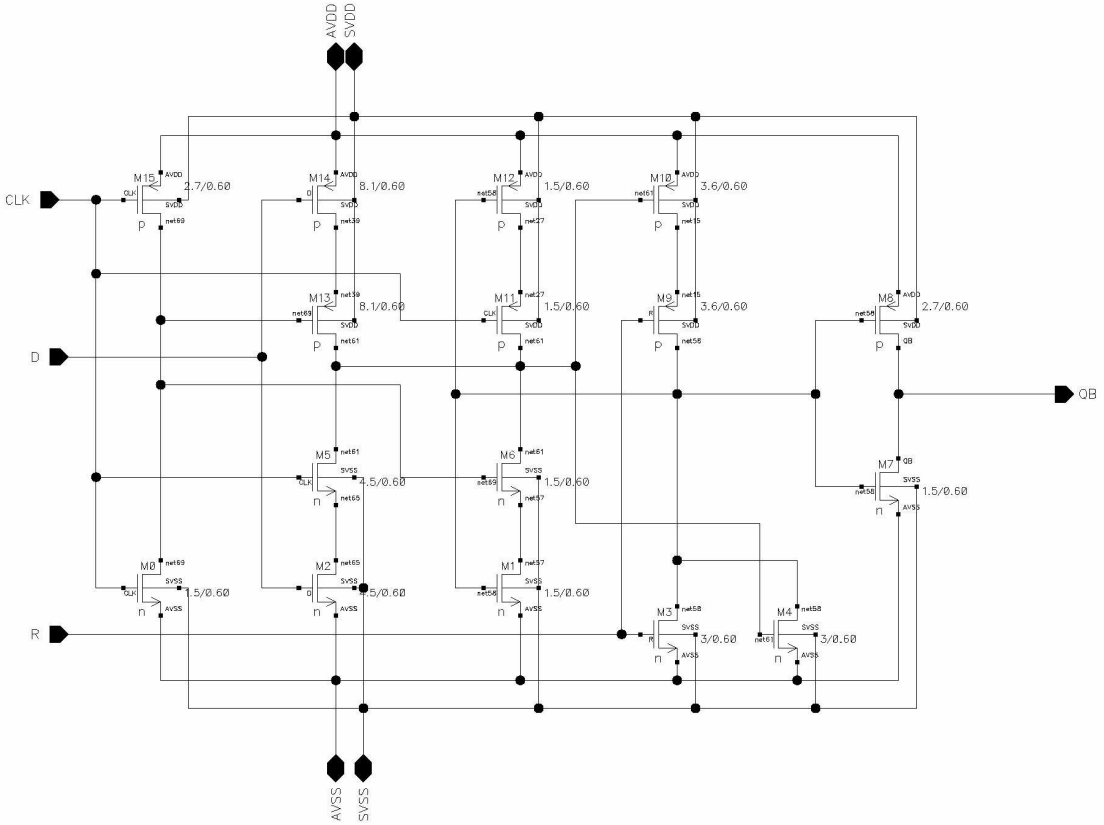
Schematic of the AND-OR circuit used in the width generator



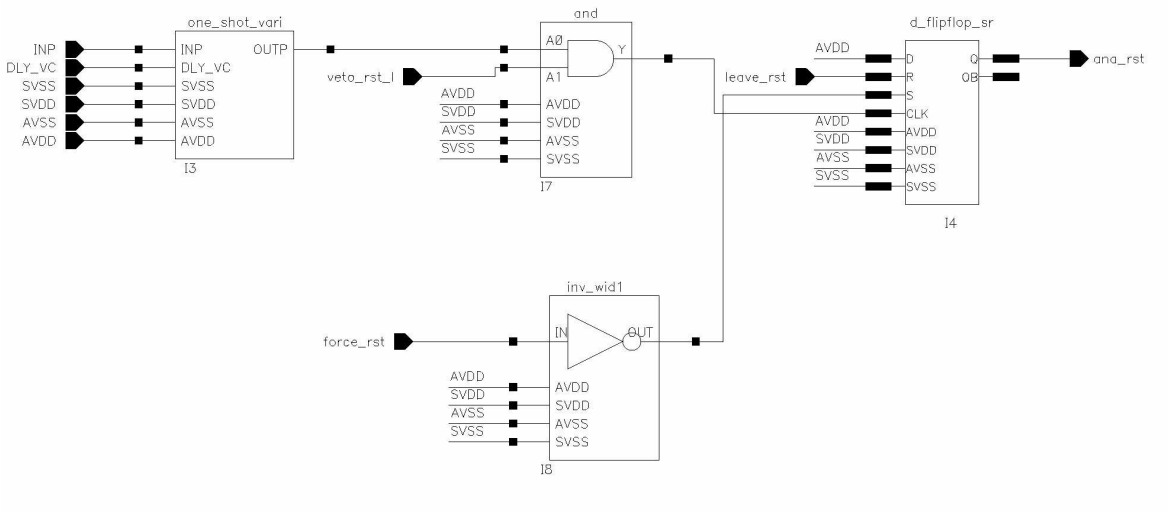
Schematic of the Inverters-OR-AND circuit used in the width generator



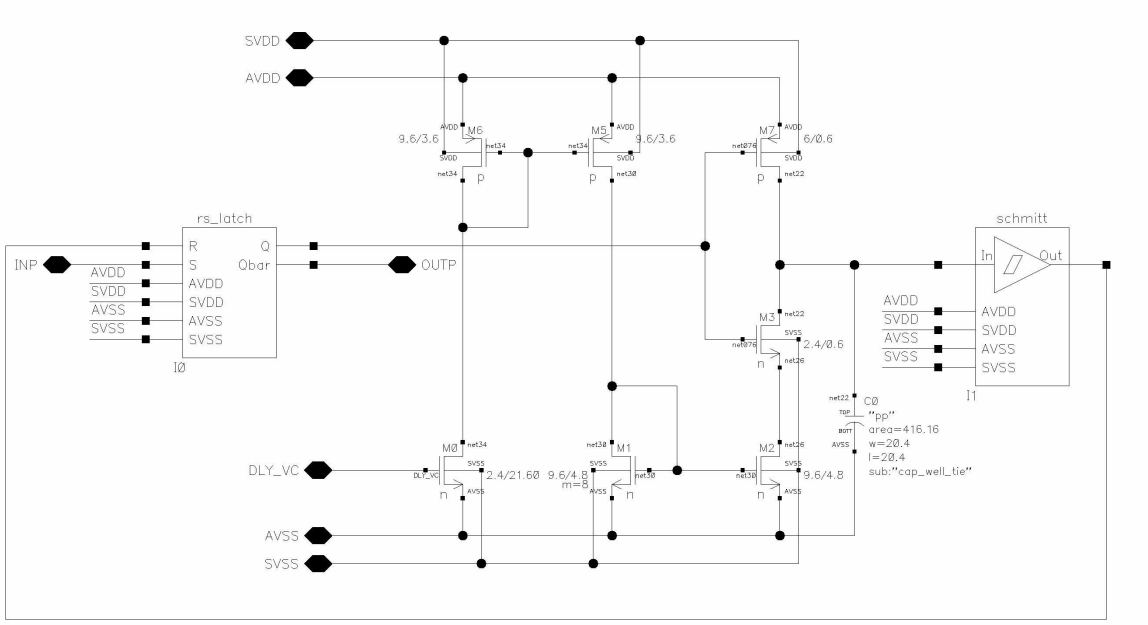
Schematic of the latch used in the width generator



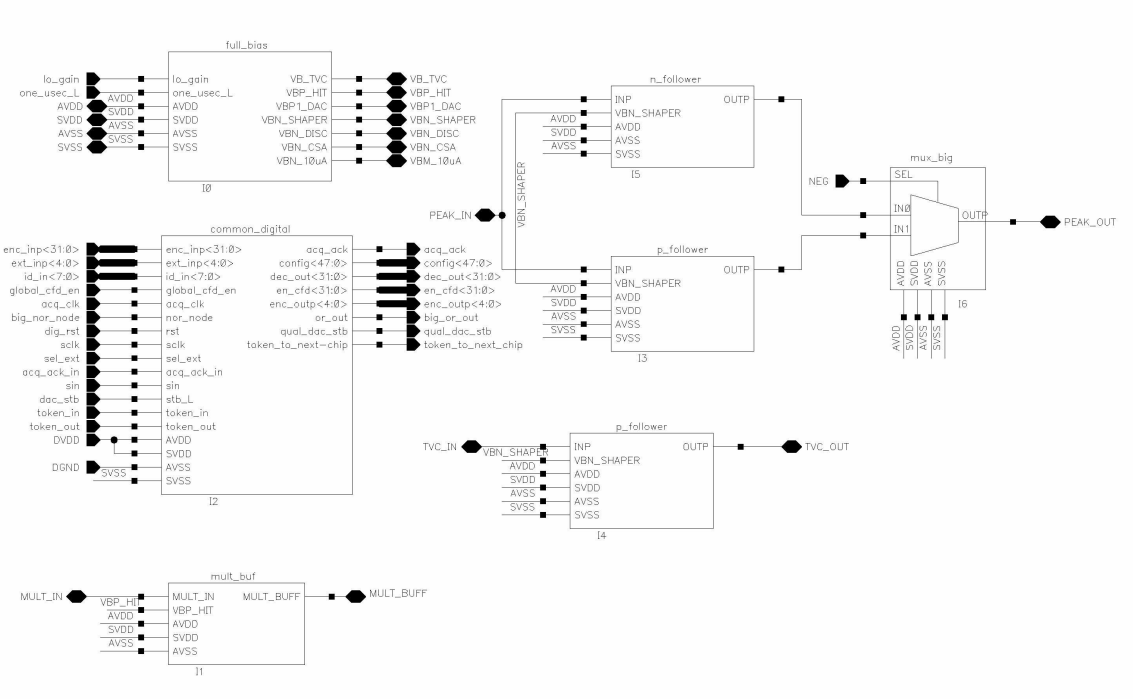
Block diagram of the reset logic



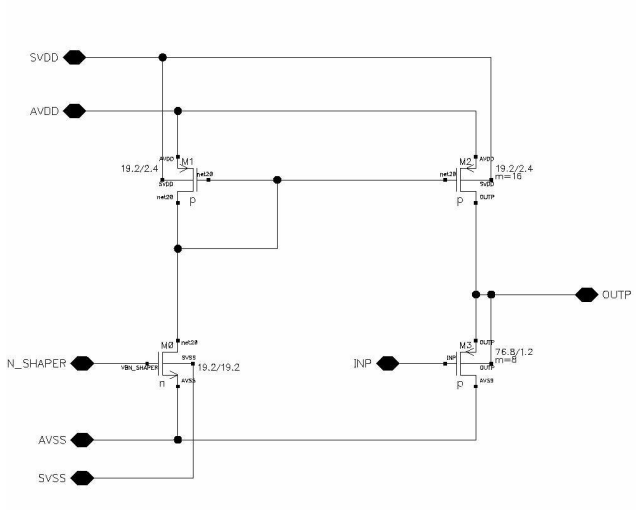
Schematic of the variable one shot circuit used in the reset logic



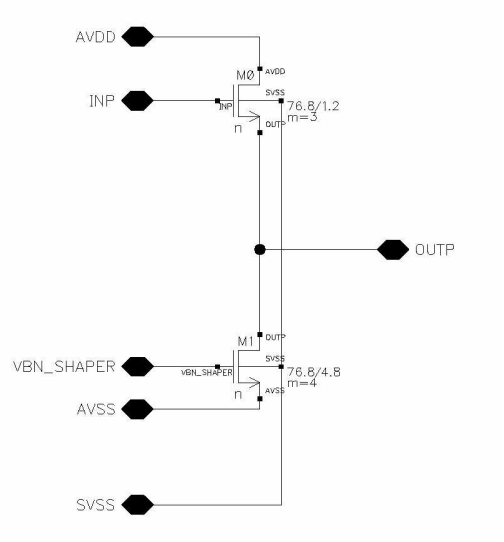
Block diagram of the common circuits



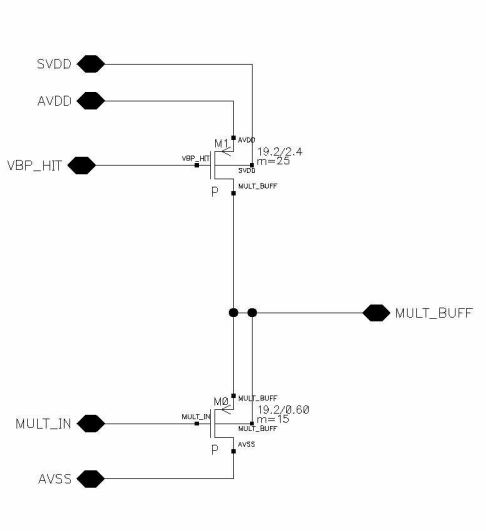
Schematic of the p-follower used in the common block



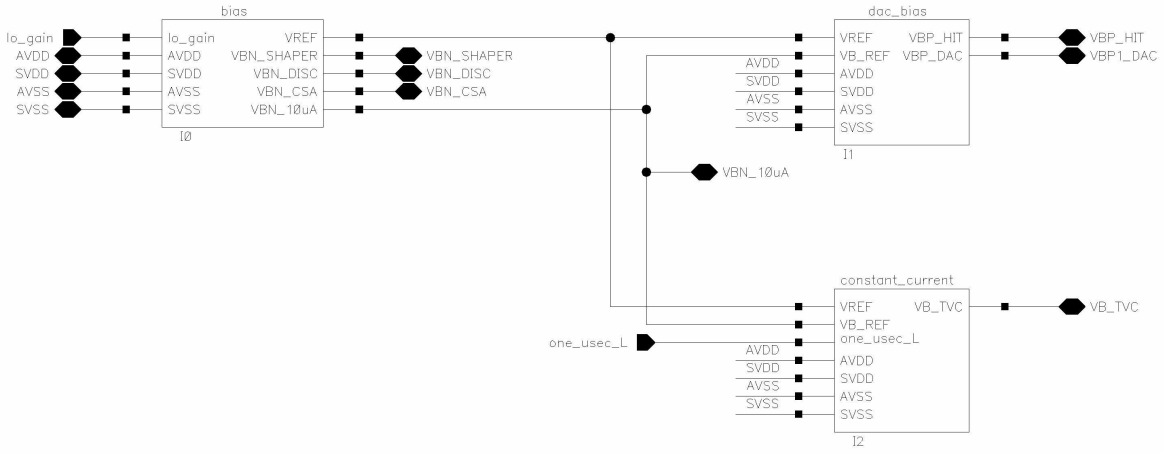
Schematic of the n-follower used in the common block



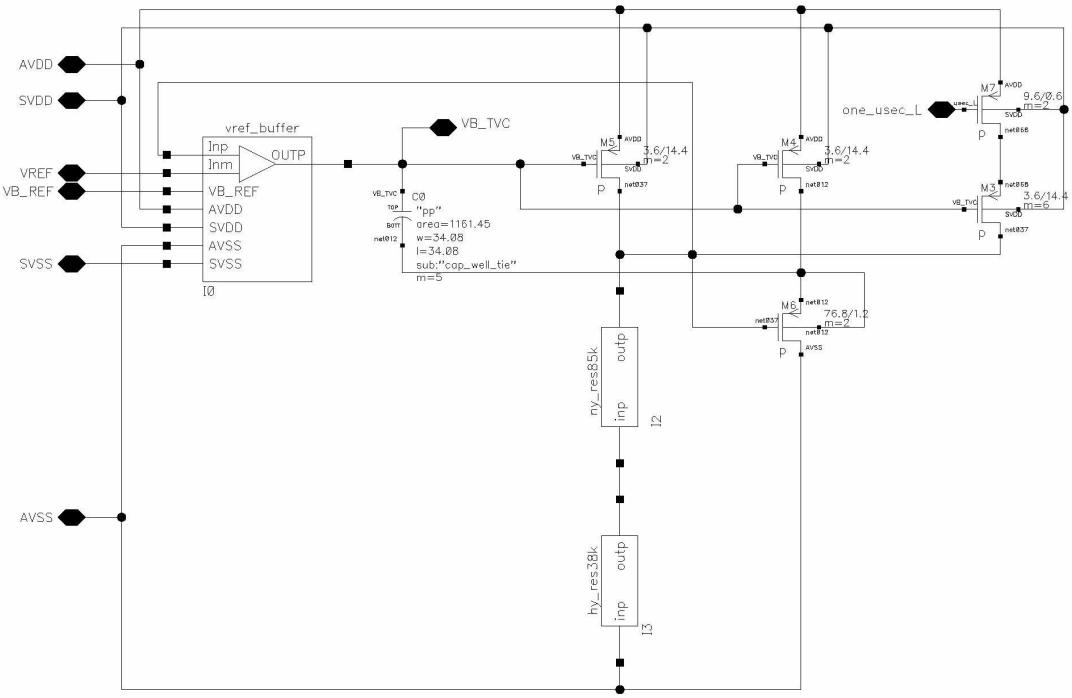
Schematic of the buffer used in the multiplicity circuit



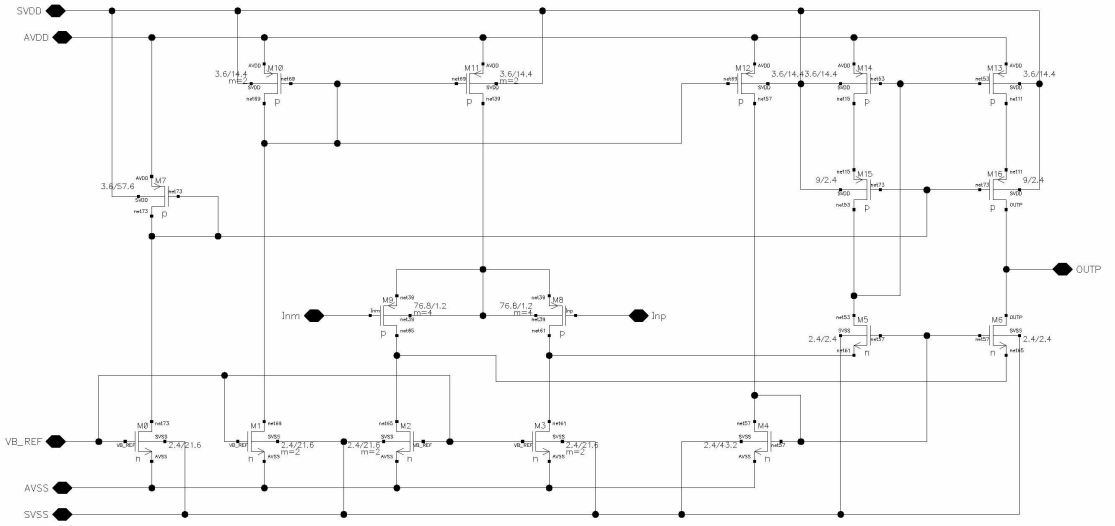
Block diagram of the bias circuits



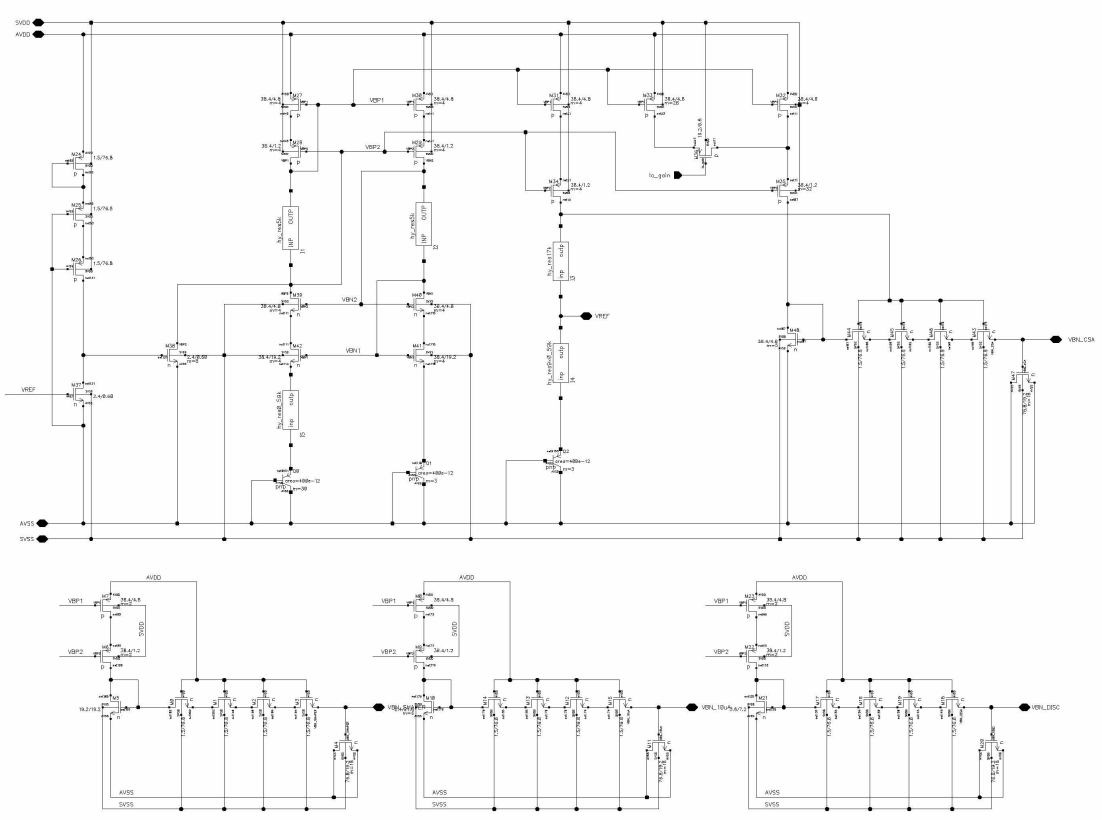
Schematic of the constant current circuit used in the bias circuits



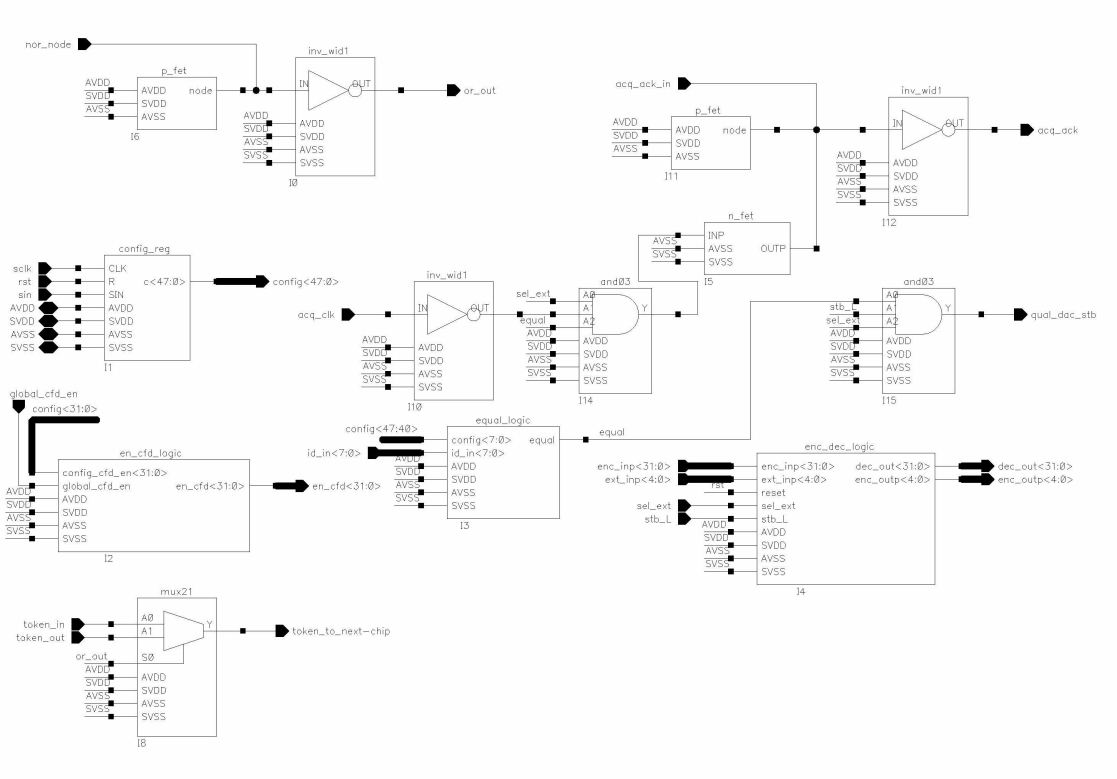
Schematic of the vref_buffer circuit used in the constant current and the dac bias circuits



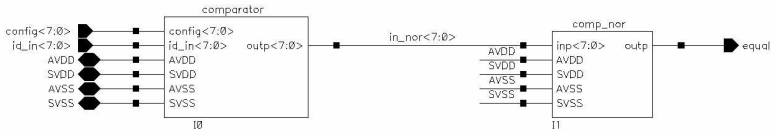
Schematic of the band gap circuit used in the bias circuits



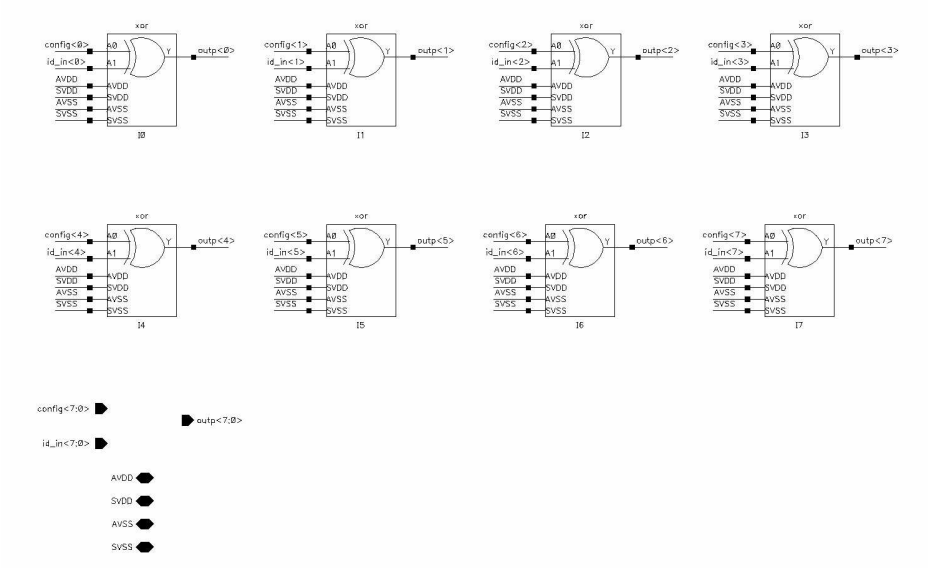
Block diagram of the common digital circuits



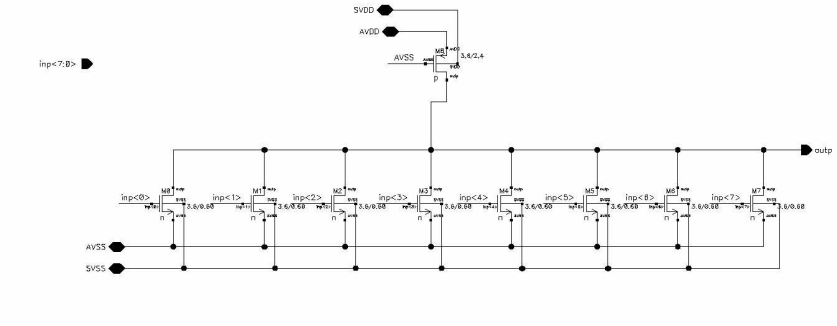
Block diagram of the equal logic used in common digital block



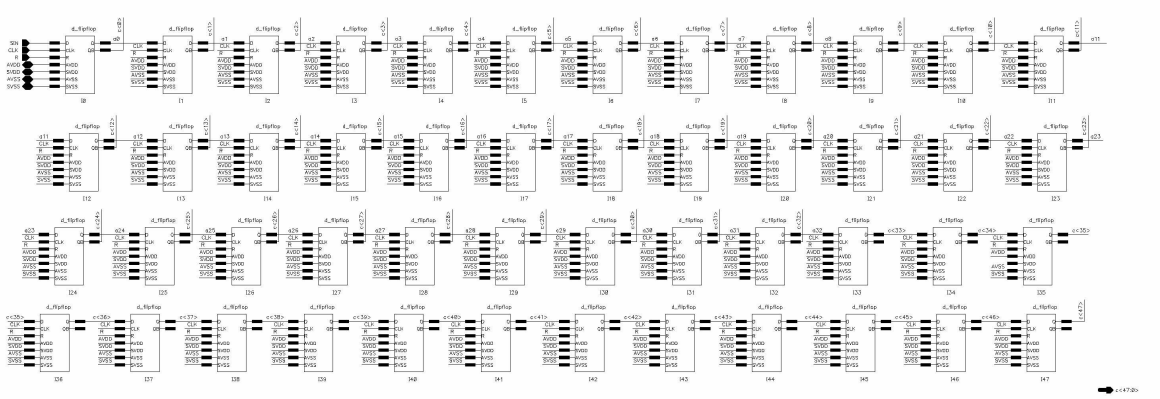
Schematic of the comparator used in the equal logic



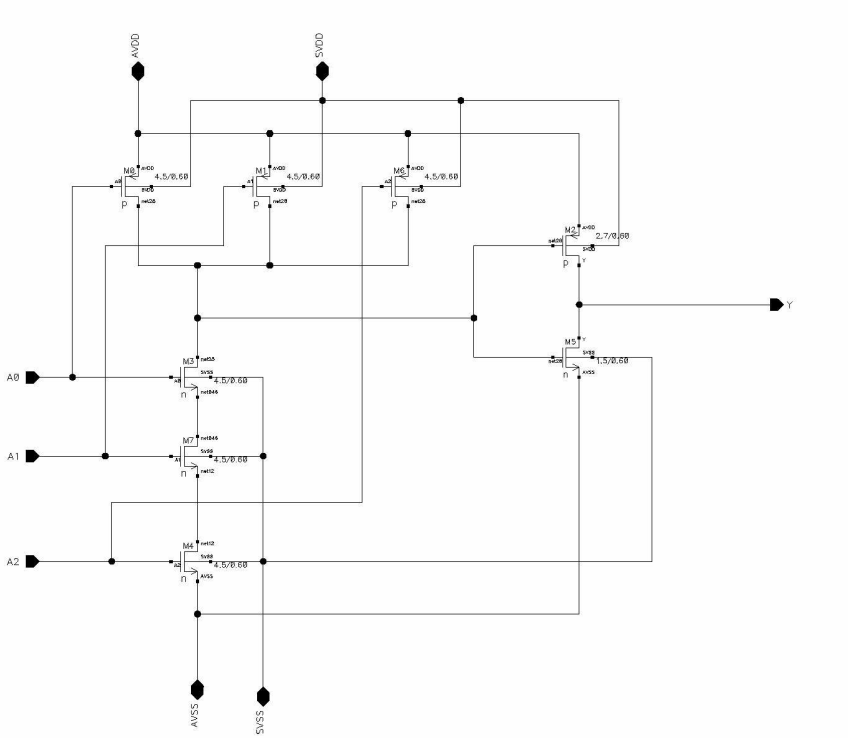
Schematic of the nor gate used in the equal logic



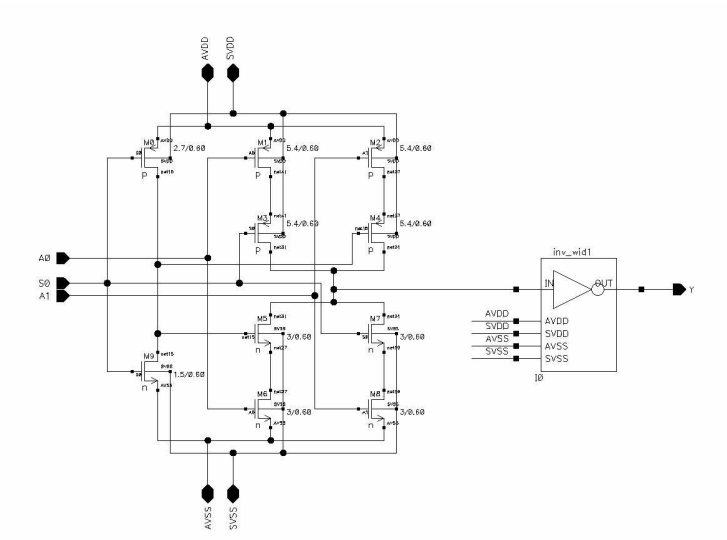
Block diagram of the configuration register



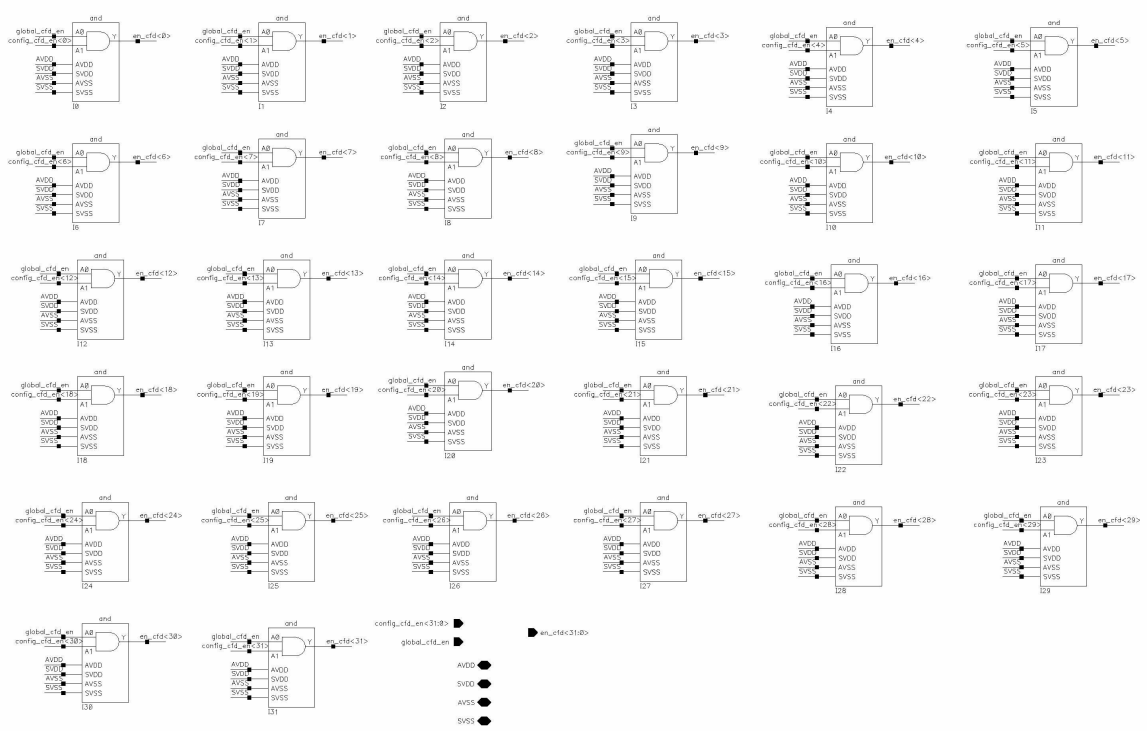
Schematic of the configuration register used in the common digital block



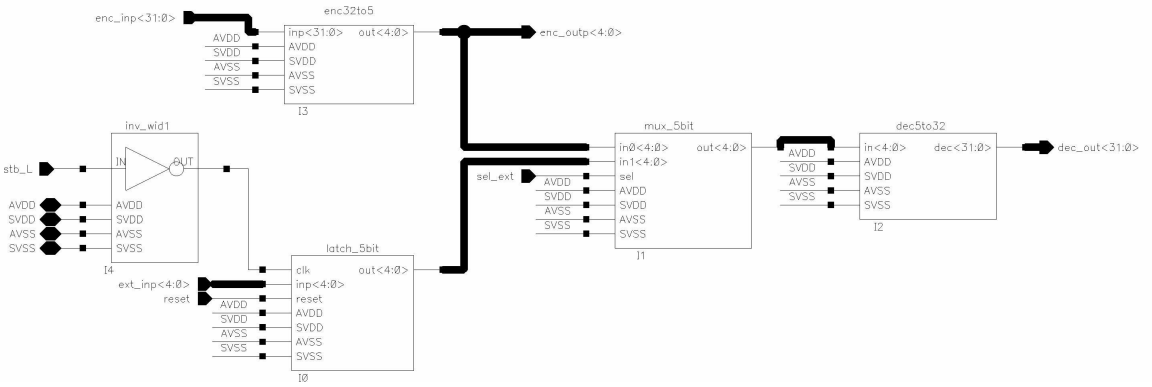
Schematic of the 2-to-1 multiplexer used in the common digital block



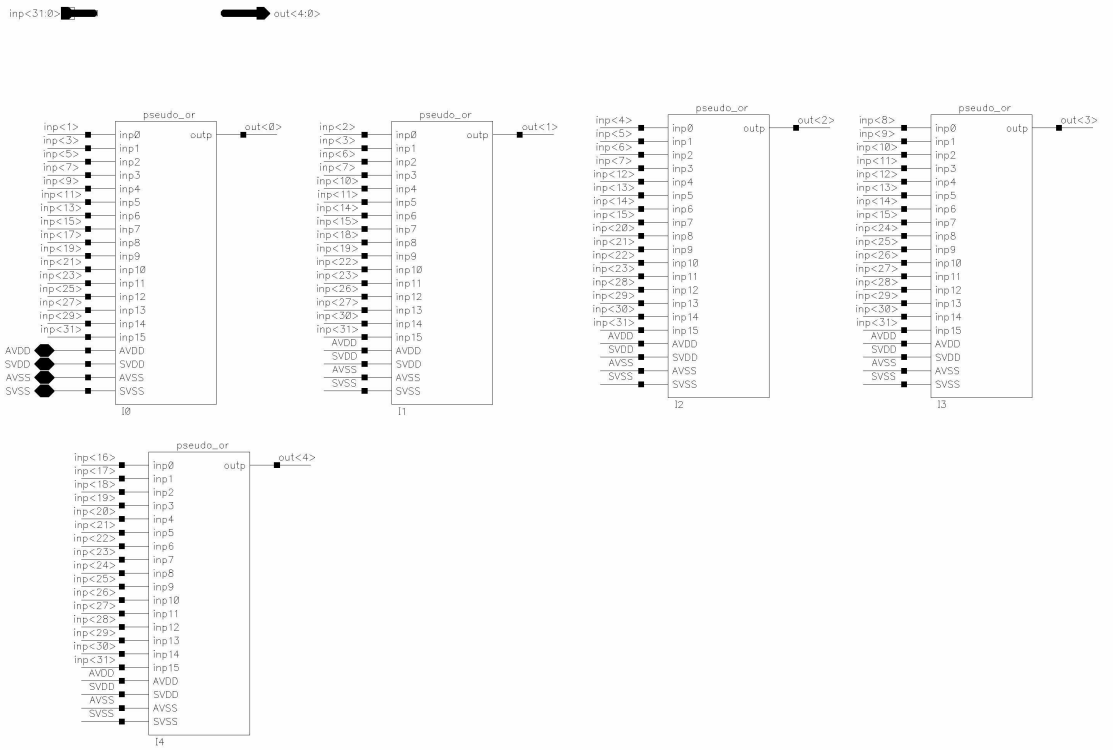
Block diagram of the enable CFD logic block



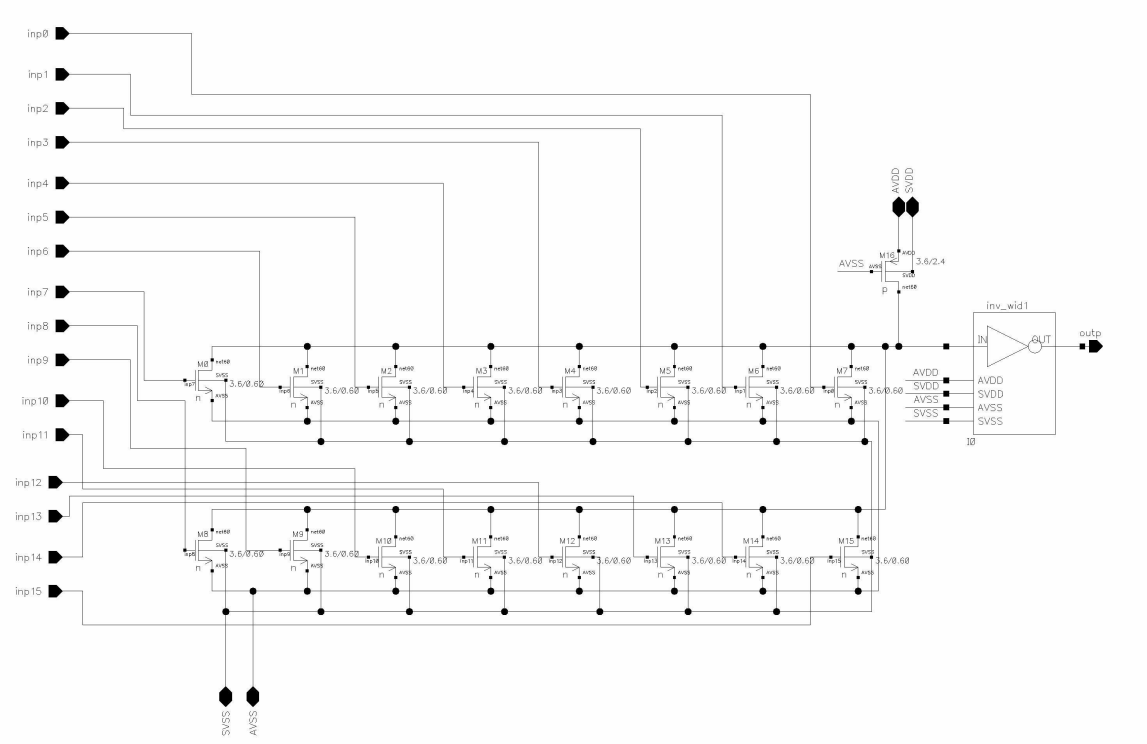
Block diagram of the channel selection block



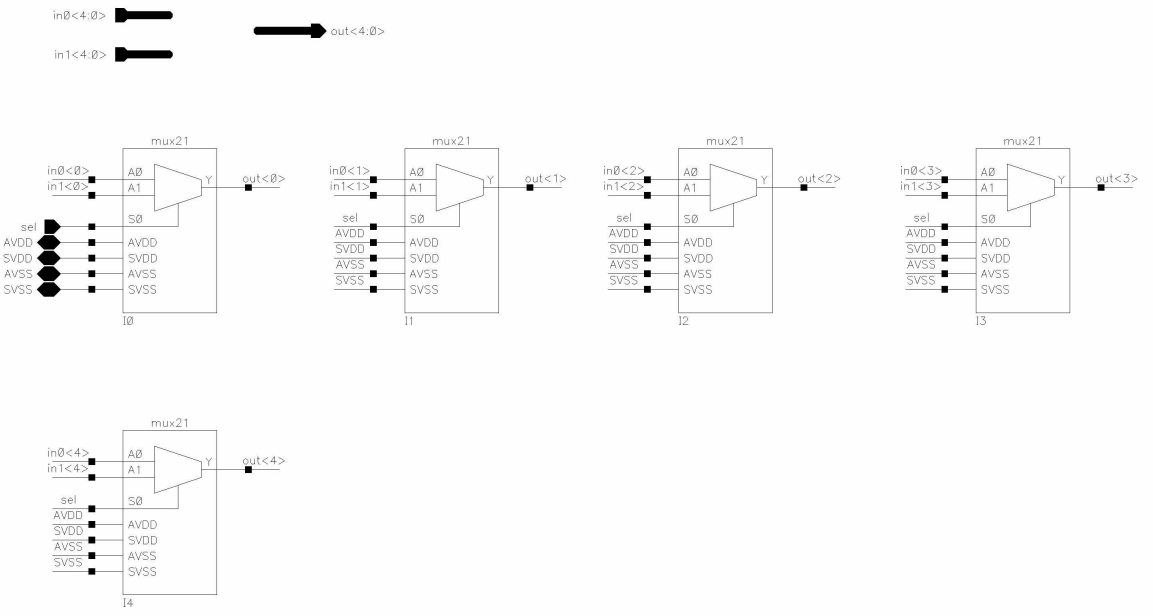
Block diagram of the 32-to-5 encoder used in the channel selection block



Schematic of the or gate used in the encoder

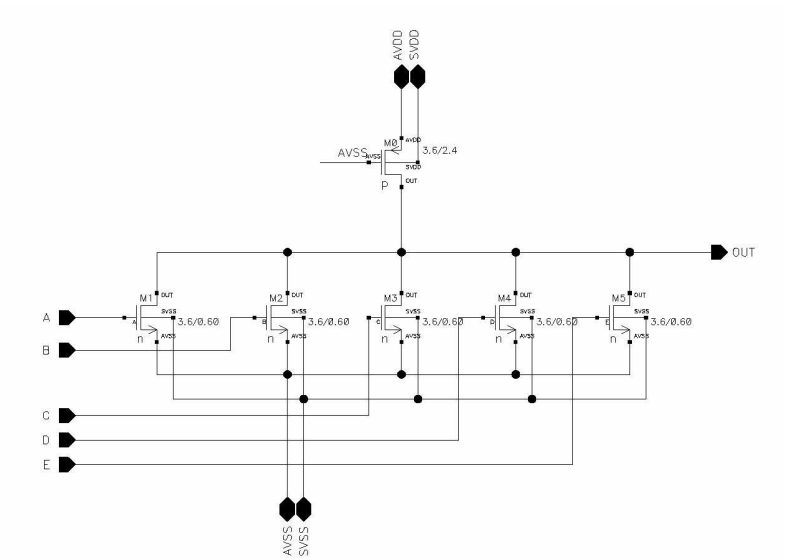


Block diagram of the 5-bit multiplexer used in the channel selection block

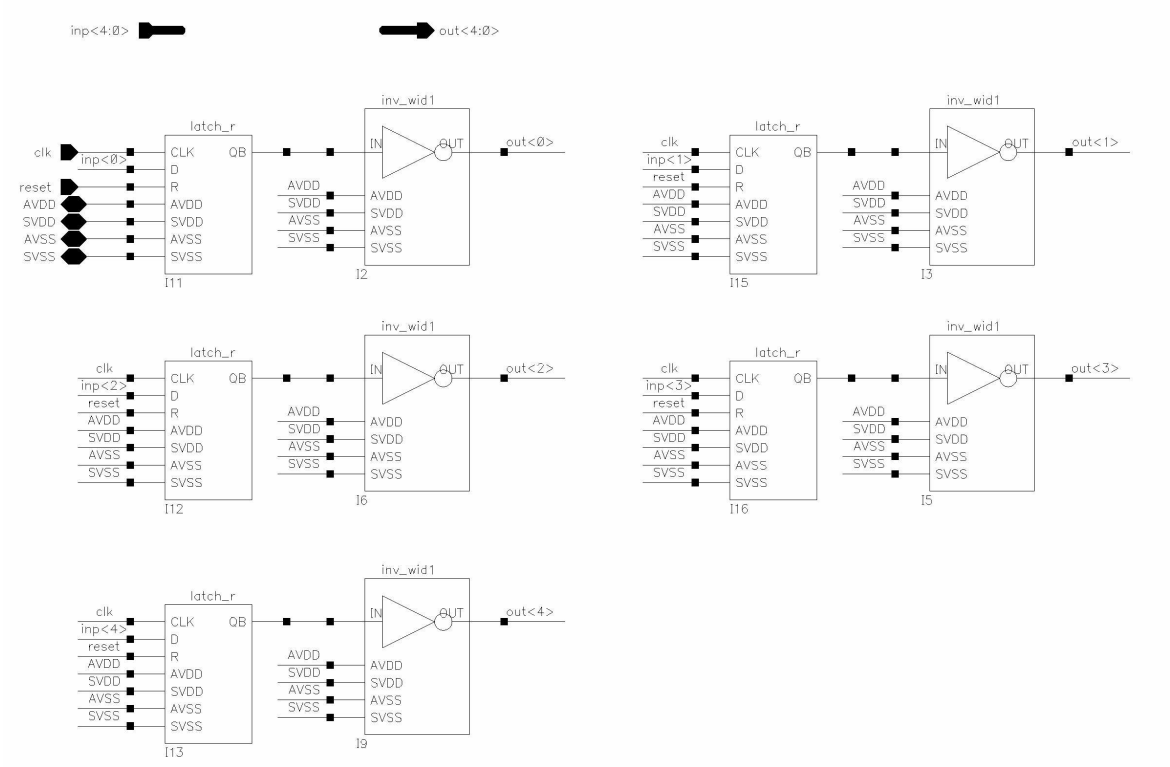


Block diagram of the 5-to32 decoder used in the channel selection block

Schematic of the nor gate used in the decoder



Block diagram of the 5-bit latch used in the channel selection block



APPENDIX C

VerilogA Codes to Perform Macro Model Simulations

```
//
// VerilogA for HinpLibMacro, and_gate
//

`include "constants.h"
`include "discipline.h"

module and_gate(Y, AVDD, AVSS, SVDD, SVSS, A0, A1);
output Y;
electrical Y;
inout AVDD, AVSS, SVDD, SVSS;
electrical AVDD, AVSS, SVDD, SVSS;
input A0, A1;
electrical A0, A1;

parameter real vlogic_high = 5;
parameter real vlogic_low = 0;
parameter real vtrans = 2.5;
parameter real tdel = 0 from [0:inf);
parameter real trise = 1p from (0:inf);
parameter real tfall = 1p from (0:inf);

real vout_val;

integer logic1, logic2;

analog begin
    logic1 = V(A0) > vtrans;
    logic2 = V(A1) > vtrans;

    @ (cross(V(A0) - vtrans, 1))  logic1 = 1;
    @ (cross(V(A0) - vtrans, -1)) logic1 = 0;
    @ (cross(V(A1) - vtrans, 1))  logic2 = 1;
    @ (cross(V(A1) - vtrans, -1)) logic2 = 0;

    //
    // define the logic function.
    //

    vout_val = (logic1 && logic2) ? vlogic_high : vlogic_low;
    V(Y) <+ transition( vout_val, tdel, trise, tfall);
end
```

```
endmodule
```

```
// VerilogA for HinpLib, 3-input and gate
//
`include "constants.h"
`include "discipline.h"

module and03(Y, AVDD, AVSS, SVDD, SVSS, A0, A1, A2);
output Y;
electrical Y;
inout AVDD, AVSS, SVDD, SVSS;
electrical AVDD, AVSS, SVDD, SVSS;
input A0, A1, A2;
electrical A0, A1, A2 ;

parameter real vlogic_high = 5;
parameter real vlogic_low = 0;
parameter real vtrans = 2.5;
parameter real tdel = 0 from [0:inf);
parameter real trise = 1p from (0:inf);
parameter real tfall = 1p from (0:inf);

real vout_val;
integer logic1, logic2, logic3;

analog begin
    logic1 = V(A0) > vtrans;
    logic2 = V(A1) > vtrans;
    logic3 = V(A2) > vtrans;

    @ (cross(V(A0) - vtrans, 1))  logic1 = 1;
    @ (cross(V(A0) - vtrans, -1)) logic1 = 0;
    @ (cross(V(A1) - vtrans, 1))  logic2 = 1;
    @ (cross(V(A1) - vtrans, -1)) logic2 = 0;
    @ (cross(V(A2) - vtrans, 1))  logic3 = 1;
    @ (cross(V(A2) - vtrans, -1)) logic3 = 0;

    //
    // define the logic function.
    //

    vout_val = (logic1 && logic2 && logic3) ? vlogic_high :
            vlogic_low;
    V(Y) <+ transition( vout_val, tdel, trise, tfall);
end
```

```
endmodule
```

```
//
// VerilogA for HinpLibMacro, d_flipflop, veriloga
//

`include "constants.h"
`include "discipline.h"

module d_flipflop(out, outb, AVDD, AVSS, SVDD, SVSS, R, clk, inp);
output out, outb;
electrical out, outb;
inout AVDD, AVSS, SVDD, SVSS;
electrical AVDD, AVSS, SVDD, SVSS;
input R, clk, inp;
electrical R, clk, inp;

parameter real vlogic_high = 5;
parameter real vlogic_low = 0;
parameter real vtrans_clk = 2.5;
parameter real vtrans = 2.5;
parameter real tdel = 0 from [0:inf);
parameter real trise = 1p from (0:inf);
parameter real tfall = 1p from (0:inf);

integer x;

analog begin
    @ (cross( V(R) - vtrans, +1 ))
       x=0;
    @ ( cross( V(clk) - vtrans_clk, +1 ))
          if(V(R) < vtrans) begin
        x = (V(inp) > vtrans);
      end
    V(out) <+ transition( vlogic_high*x + vlogic_low*!x,tdel, trise,
                    tfall );
    V(outb) <+ transition( vlogic_high*!x + vlogic_low*x,tdel, trise,
                     tfall );
end
endmodule
```

```
//
// VerilogA for HinpLibMacro, d_flipflop_sr
//
```

```
`include "constants.h"
`include "discipline.h"

module d_flipflop_sr(out, outb, AVDD, AVSS, SVDD, SVSS, R, S, clk, inp);
output out, outb;
electrical out, outb;
inout AVDD, AVSS, SVDD, SVSS;
electrical AVDD, AVSS, SVDD, SVSS;
input R, S, clk, inp;
electrical R S, clk, inp;

parameter real vlogic_high = 5;
parameter real vlogic_low = 0;
parameter real vtrans_clk = 2.5;
parameter real vtrans = 2.5;
parameter real tdel = 0 from [0:inf);
parameter real trise = 1p from (0:inf);
parameter real tfall = 1p from (0:inf);

integer x;

analog begin
    @ (cross( V(R) - vtrans, +1 ))
        x=0;
    @ (cross( V(S) - vtrans, -1 ))
        x=1;
    @ (cross( V(clk) - vtrans_clk, +1 ))
            if((V(R) < vtrans ) && (V(S) > vtrans)) begin
               x = (V(inp) > vtrans);
            end
     V(out) <+ transition( vlogic_high*x + vlogic_low*!x,tdel, trise, tfall );

     V(outb) <+ transition( vlogic_high*!x + vlogic_low*x,tdel, trise,  tfall );
end
endmodule
```

```
//
// VerilogA for HinpLib, dac
//

`include "constants.h"
`include "discipline.h"

module dac(DAC_OUT, OUT_N, OUT_P, AVDD, AVSS, SVDD, SVSS, VBP1_DAC,
```

```
                dac);
output DAC_OUT, OUT_N, OUT_P;
electrical DAC_OUT, OUT_N, OUT_P;
inout AVDD, AVSS, SVDD, SVSS;
electrical AVDD, AVSS, SVDD, SVSS;
input VBP1_DAC;
electrical VBP1_DAC;
input [5:0] dac;
electrical [5:0] dac;

parameter real Vth = 2.5;
parameter real tr = 1n from [0:inf);
parameter real tf = 1n from [0:inf);
parameter real Ron = 1k ;
parameter real Roff = 1e12 ;
parameter real iscale = 5u from [0:inf);

real     value_of_current, del [4:0], outp_voltage, Rch_n, Rch_p, pow2 [4:0] ;

integer   i ;

analog begin
  @(initial_step) begin
    pow2[4] = 16 ;
    pow2[3] = 8 ;
    pow2[2] = 4 ;
    pow2[1] = 2 ;
    pow2[0] = 1 ;
  end ;
  if (analysis("static")) begin
    Rch_n = Roff ;
    Rch_p = Roff ;
    value_of_current = 0.0 ;
  end else
  begin
    value_of_current = 0.0 ;
    if (V(dac[0]) < Vth) del[0] = pow2[0] ;
      else  del[0] = 0 ;
      if (V(dac[1]) < Vth) del[1] = pow2[1] ;
      else  del[1] = 0 ;
      if (V(dac[2]) < Vth) del[2] = pow2[2] ;
      else  del[2] = 0 ;
      if (V(dac[3]) < Vth) del[3] = pow2[3] ;
      else  del[3] = 0 ;
      if (V(dac[4]) < Vth) del[4] = pow2[4] ;
      else  del[4] = 0 ;
```

```
        for (i=0; i<5; i=i+1) begin
      value_of_current = value_of_current + del[i] ;
        end;
    if (V(dac[5]) < Vth) begin
      Rch_n = Ron ;
      Rch_p = Roff ;
    end
    else begin
      Rch_n = Roff ;
      Rch_p = Ron ;
    end ;
  end ;
  I(OUT_N, DAC_OUT) <+ V(OUT_N, DAC_OUT) / transition(Rch_n, 0, tr, tf) ;
  I(OUT_P, DAC_OUT) <+ V(OUT_P, DAC_OUT) / transition(Rch_p, 0, tr, tf) ;
  I(OUT_P) <+ -1 * transition(iscale * value_of_current, 0, tr, tf);
  I(OUT_N) <+  transition(iscale * value_of_current, 0, tr, tf);
end
endmodule
```

```
//
// VerilogA for HinpLibMacro, diff_amp_le
//

`include "constants.h"
`include "discipline.h"

module diff_amp_le(AVDD, AVSS, INM, INP, OUTM, OUTP, SVDD, SVSS,
                   VBN_DISC);
inout AVDD, AVSS, SVDD, SVSS;
electrical AVDD, AVSS, SVDD, SVSS;
inout INM, INP, OUTM, OUTP, VBN_DISC;
electrical INM, INP, OUTM, OUTP, VBN_DISC;

parameter  real  A0 = 10 ;
parameter  real  vcmo = 3 ;
parameter  real  w0 = 2 * 3.14 * 100e6 ;

real   diff_out_volt ;

analog begin
//
// Gain of A0, 3 db BW of f0, and common mode output level
// of cmo_level
//
  @(initial_step) diff_out_volt = 0 ;
```

```
  if (analysis("static")) begin
        diff_out_volt = 0 ;
        V(OUTP) <+ vcmo ;
        V(OUTM) <+ vcmo ;
   end
   diff_out_volt = laplace_nd( V(INP, INM),{A0, 0}, {1, 1/w0} ) ;
   V(OUTP) <+ vcmo + diff_out_volt / 2  ;
   V(OUTM) <+ vcmo - diff_out_volt / 2  ;
end
endmodule
```

```
//
// VerilogA for HinpLib, full_bias
//

`include "constants.h"
`include "discipline.h"

module full_bias(AVDD, AVSS, SVDD, SVSS, VBN_10uA, VBN_CSA, VBN_DISC,
                 VBN_SHAPER, VBP1_DAC, VBP_HIT, VB_TVC,lo_gain, one_usec_L);
inout AVDD, AVSS, SVDD, SVSS;
electrical AVDD, AVSS, SVDD, SVSS;
inout VBN_10uA, VBN_CSA, VBN_DISC, VBN_SHAPER, VBP1_DAC, VBP_HIT,
      VB_TVC;
electrical VBN_10uA, VBN_CSA, VBN_DISC, VBN_SHAPER, VBP1_DAC, VBP_HIT,
           VB_TVC;
input lo_gain, one_usec_L;
electrical lo_gain, one_usec_L;

parameter real logic_thresh = 2.5 ;

analog  begin
  V(VBN_SHAPER)  <+ 1.8519 ;
  V(VBN_DISC)  <+ 2.6245 ;
  V(VBN_10uA) <+ 1.9829 ;
  if (V(lo_gain) > logic_thresh) V(VBN_CSA)  <+ 0.91164 ;
  else V(VBN_CSA)  <+ 1.3662 ;
  V(VBP1_DAC)  <+ 3.3796 ;
  V(VBP_HIT)  <+ 3.4597 ;
  V(VB_TVC)  <+ 3.3650 ;
end
endmodule
```

```
//
// VerilogA for HinpLib, full_tvc
//

`include "constants.h"
`include "discipline.h"

module full_tvc(AVDD, AVSS, CAP_GND, SVDD, SVSS, VB_TVC, ana_reset, dig_reset,
                start, stop_tvc, tvc_out);
inout AVDD, AVSS, SVDD, SVSS;
electrical AVDD, AVSS, SVDD, SVSS;
inout CAP_GND, VB_TVC, ana_reset, dig_reset, start, stop_tvc, tvc_out;
electrical CAP_GND, VB_TVC, ana_reset, dig_reset, start, stop_tvc, tvc_out;

parameter   real   Vth = 2.5 from (1 : 4) ;
parameter   real   logic_low = 0 ;
parameter   real   logic_high = 5 ;
parameter   real   vscale = 1 ;
parameter   real   gmi = 10e-6 ;
parameter   real   tr = 1n ;
parameter   real   tf = 1n ;
parameter   real   vos = 1 from (0.5 : 2.5) ;

real       ramp_voltage, tscale, run_status, offset ;

analog begin
   @(initial_step) begin
         ramp_voltage = 0 ;
         run_status = logic_low;
    end ;
    @(cross(V(start)>Vth)) begin
        offset = $realtime ;
         run_status = logic_high ;
     end ;
     if (V(VB_TVC) > 0.5) tscale = 250n ;
     else tscale = 1u ;
     if (V(dig_reset)>Vth) run_status = logic_low ;
     if (V(stop_tvc)>Vth) run_status = logic_low ;
     if (V(ana_reset) > Vth) ramp_voltage = 0 ;
     else if (run_status > Vth)
           ramp_voltage = vscale * ($realtime - offset) / tscale ;
     if (ramp_voltage > 4) ramp_voltage = 4 ;
     I(VB_TVC) <+ gmi * V(VB_TVC) ;
     V(tvc_out) <+ transition(ramp_voltage, 0, tr, tf) + vos ;
end
endmodule
```

```
//
// VerilogA for HinpLib, comp_nor
//
`include "constants.h"
`include "discipline.h"

module comp_nor(outp, AVDD, AVSS, SVDD, SVSS, inp);
output outp;
electrical outp;
inout AVDD, AVSS, SVDD, SVSS;
electrical AVDD, AVSS, SVDD, SVSS;
input [7:0] inp;
electrical [7:0] inp;

parameter real vlogic_high = 5;
parameter real vlogic_low = 0;
parameter real vtrans = 2.5;
parameter real tdel = 0 from [0:inf);
parameter real trise = 1p from (0:inf);
parameter real tfall = 1p from (0:inf);

real vout_val;
integer logic0,logic1,logic2,logic3,logic4,logic5, logic6,logic7;

analog begin
    logic0 = V(inp[0]) > vtrans;
    logic1 = V(inp[1]) > vtrans;
    logic2 = V(inp[2]) > vtrans;
    logic3 = V(inp[3]) > vtrans;
    logic4 = V(inp[4]) > vtrans;
    logic5 = V(inp[5]) > vtrans;
    logic6 = V(inp[6]) > vtrans;
    logic6 = V(inp[7]) > vtrans;
    @ (cross(V(inp[0]) - vtrans, 1))  logic0 = 1;
    @ (cross(V(inp[0]) - vtrans, -1)) logic0 = 0;
    @ (cross(V(inp[1]) - vtrans, 1))  logic1 = 1;
    @ (cross(V(inp[1]) - vtrans, -1)) logic1 = 0;
    @ (cross(V(inp[2]) - vtrans, 1))  logic2 = 1;
    @ (cross(V(inp[2]) - vtrans, -1)) logic2 = 0;
    @ (cross(V(inp[3]) - vtrans, 1))  logic3 = 1;
    @ (cross(V(inp[3]) - vtrans, -1)) logic3 = 0;
    @ (cross(V(inp[4]) - vtrans, 1))  logic4 = 1;
    @ (cross(V(inp[4]) - vtrans, -1)) logic4 = 0;
    @ (cross(V(inp[5]) - vtrans, 1))  logic5 = 1;
```

```
    @ (cross(V(inp[5]) - vtrans, -1)) logic5 = 0;
    @ (cross(V(inp[6]) - vtrans, 1))  logic6 = 1;
    @ (cross(V(inp[6]) - vtrans, -1)) logic6 = 0;
    @ (cross(V(inp[7]) - vtrans, 1))  logic7 = 1;
    @ (cross(V(inp[7]) - vtrans, -1)) logic7 = 0;


    //
    // define the logic function.
    //
    vout_val = (!(logic0 || logic1 || logic2 || logic3 ||
            logic4 || logic5 || logic6 || logic7 )) ? vlogic_high : vlogic_low;

    V(outp) <+ transition( vout_val, tdel, trise, tfall);
end
endmodule
```

```
//
// VerilogA for HinpLib, csa
//

`include "constants.h"
`include "discipline.h"

module csa(AVDD, AVSS, CSA_GND, INP, OUTP, SVDD, SVSS, VBN_CSA,
            LO_GAIN);
inout AVDD, AVSS, SVDD, SVSS;
electrical AVDD, AVSS, SVDD, SVSS;
inout CSA_GND, INP, OUTP, VBN_CSA;
electrical CSA_GND, INP, OUTP, VBN_CSA;
input LO_GAIN;
electrical LO_GAIN;

parameter   real   gm_lo = 2.4e-3 ;
parameter   real   gm_hi = 12e-3 ;
parameter   real   Gf_lo = 500e-9 ;
parameter   real   Gf_hi = 100e-9 ;
parameter   real   Cf_lo = 12.5e-12 ;
parameter   real   Cf_hi = 2.5e-12 ;
parameter   real   Cfl_lo = 14e-12 ;
parameter   real   Cfl_hi = 3e-12 ;
parameter   real   Ct_lo = 95e-12 ;
parameter   real   Ct_hi = 85e-12 ;
parameter   real   b0_lo = gm_lo * Gf_lo ;
parameter   real   b0_hi = gm_hi * Gf_hi ;
```

```
parameter    real   b1_lo = gm_lo * Cf_lo ;
parameter    real   b1_hi = gm_hi * Cf_hi ;
parameter    real   b2_lo = Ct_lo * Cfl_lo ;
parameter    real   b2_hi = Ct_hi * Cfl_hi ;
parameter    real   Vtp = 0.75 ;
parameter    real   Vdss = 0.48546 ;

real    Vin0, Vin1, vos ;

analog begin
      @(initial_step) begin
            vos = V(CSA_GND) - Vtp - Vdss ;
      end ;
      if (analysis("static")) begin
            vos = V(CSA_GND) - Vtp - Vdss ;
            Vin0 = vos ;
            Vin1 = vos ;
            V(OUTP) <+ vos ;
      end
      Vin0 = vos + 0.9 * laplace_nd(I(INP), {-1 * gm_hi, 0, 0}, {b0_hi, b1_hi, b2_hi}) ;
      Vin1 = vos + 0.9 * laplace_nd(I(INP), {-1 * gm_lo , 0, 0}, {b0_lo, b1_lo , b2_lo}) ;
      if (V(LO_GAIN) == 0)
               V(OUTP) <+ Vin0 ;
      else
               V(OUTP) <+ Vin1 ;
end
endmodule
```

```
//
// VerilogA for HinpLib, cmprtr, veriloga
//

`include "constants.h"
`include "discipline.h"

module cmprtr(OUTP, AVDD, AVSS, INM, INP, SVDD, SVSS, VBN_DISC);
output OUTP;
electrical OUTP;
inout AVDD, AVSS, SVDD, SVSS;
electrical AVDD, AVSS, SVDD, SVSS;
inout INM, INP, VBN_DISC;
electrical INM, INP, VBN_DISC ;

parameter real output_high = 5;
parameter real output_low = 0 ;
```

```
parameter real slope = 1e4 from (0:inf);

real    high_minus_low, high_plus_low ;

analog begin
      if (analysis("static")) begin
        @(initial_step) begin
          high_minus_low = 0.5*(output_high - output_low);
          high_plus_low  = 0.5*(output_high + output_low);
        end
      end
      V(OUTP) <+ high_minus_low * tanh(slope*V(INP, INM)) + high_plus_low;
end
endmodule
```

```
//
// VerilogA for HinpLib, gain_amp, veriloga
//

`include "constants.h"
`include "discipline.h"

module gain_amp(AGND, AVDD, AVSS, INP, OUTP, SVDD, SVSS, VB_SHAPER);
inout AGND, INP, OUTP, VB_SHAPER;
electrical AGND, INP, OUTP, VB_SHAPER;
inout AVDD, AVSS, SVDD, SVSS;
electrical AVDD, AVSS, SVDD, SVSS;

 parameter    real   gain = 4.0 ;

 real   vout ;

analog begin
      @(initial_step) begin
            vout =  V(AGND) ;
      end ;
      if (analysis("static")) begin
            vout = V(AGND) ;
      end
      else
      begin
            vout = V(AGND) - gain * (V(INP) - V(AGND))  ;
      end ;
      V(OUTP) <+ vout ;
end
```

```
endmodule
```

```
//
// VerilogA for HinpLib, inv_12
//

`include "constants.h"
`include "discipline.h"

module inv_12(Y, A, AVDD, AVSS, SVDD, SVSS);
output Y;
electrical Y;
input A;
electrical A;
inout AVDD, AVSS, SVDD, SVSS;
electrical AVDD, AVSS, SVDD, SVSS;

parameter real tdel = 3n from [0:inf);
parameter real trise = 1p from (0:inf);
parameter real tfall = 1p from (0:inf);

analog begin
    V(Y) <+ transition(V(A), tdel, trise, tfall);
end
endmodule
```

```
//
// VerilogA for HinpLib, inv_wid1, veriloga
//

`include "constants.h"
`include "discipline.h"

module inv_wid1(OUT, AVDD, AVSS, IN, SVDD, SVSS);
output OUT;
electrical OUT;
inout AVDD, AVSS, SVDD, SVSS;
electrical AVDD, AVSS, SVDD, SVSS;
input IN;
electrical IN;

parameter real vlogic_high = 5;
parameter real vlogic_low = 0;
```

```
parameter real vtrans = 2.5;
parameter real tdel = 0 from [0:inf);
parameter real trise = 1p from (0:inf);
parameter real tfall = 1p from (0:inf);


real X;

integer logic_in;

analog begin
        logic_in = V(IN) > vtrans;
        @ (cross(V(IN) - vtrans, 1))  logic_in = 1;
        @ (cross(V(IN) - vtrans, -1)) logic_in = 0;

        //
        // define the logic function.
        //
        X = !(logic_in) ? vlogic_high : vlogic_low;
       V(OUT) <+ transition( X, tdel, trise, tfall);
end
endmodule
```

```
//
// VerilogA for HinpLibMacro, latch_r
//

`include "constants.h"
`include "discipline.h"

module latch_r(QB, AVDD, AVSS, SVDD, SVSS, R, clk, inp);
output QB;
electrical QB;
inout AVDD, AVSS, SVDD, SVSS;
electrical AVDD, AVSS, SVDD, SVSS;
input R, clk, inp;
electrical R, clk, inp;

parameter real vlogic_high = 5;
parameter real vlogic_low = 0;
parameter real vtrans = 2.5;
parameter real tdel = 0 from [0:inf);
parameter real trise = 1p from (0:inf);
parameter real tfall = 1p from (0:inf);
```

```
integer x;

analog begin
  if(V(clk)> vtrans)
     x = (V(inp) > vtrans);
  else
     x=x;
  V(QB) <+ transition( vlogic_high*!x + vlogic_low*x, tdel, trise, tfall );
end
endmodule
```

```
//
// VerilogA for HinpLibMacro, mult_buf
//

`include "constants.h"
`include "discipline.h"

module mult_buf(AVDD, AVSS, SVDD, mult_in, mult_out, vbphit);
inout AVDD, AVSS, SVDD;
electrical AVDD, AVSS, SVDD;
inout mult_in, mult_out, vbphit;
electrical mult_in, mult_out, vbphit;
electrical x;

parameter  real  Rout = 1k ;
parameter  real  Vtp = 0.95 ;

analog begin
     V(x) <+ V(mult_in) + Vtp ;
     I(x, mult_out) <+ V(x, mult_out) / Rout ;
end
endmodule
```

```
//
// VerilogA for HinpLib, mux21, veriloga
//

`include "constants.h"
`include "discipline.h"
```

```
module mux21(Y, AVDD, AVSS, SVDD, SVSS, A0, A1, S0);
output Y;
electrical Y;
inout AVDD, AVSS, SVDD, SVSS;
electrical AVDD, AVSS, SVDD, SVSS;
input A0, A1, S0;
electrical A0, A1, S0;

parameter real vtrans = 2.5;
parameter real tdel = 0 from [0:inf);
parameter real trise = 1p from (0:inf);
parameter real tfall = 1p from (0:inf);

real X;

analog begin
    if( V(S0) > vtrans)
        X =(V(A1) > vtrans);
    else
      X=(V(A0) > vtrans);
    V(Y) <+ transition(X, tdel, trise, tfall);
end
endmodule
```

```
//
// VerilogA for HinpLib, mux_big
//

`include "constants.h"
`include "discipline.h"

module mux_big(AVDD, AVSS, IN0, IN1, OUTP, SVDD, SVSS, SEL);
inout AVDD, AVSS, SVDD, SVSS;
electrical AVDD, AVSS, SVDD, SVSS;
inout IN0, IN1, SEL, OUTP;
electrical IN0, IN1, SEL, OUTP;

parameter   real Vth=2.5 from (0:5) ;
parameter   real Ron = 200 from (0:inf) ;
parameter   real Roff = 1e12 from (0:inf) ;
parameter   real tr=10e-12 from [0:inf) ;
parameter   real tf=10e-12 from [0:inf) ;

real Rch0, Rch1 ;
```

```
analog begin
    if(V(SEL) > Vth) begin
        Rch1 = Ron ;
        Rch0 = Roff ;
    end
    else begin
        Rch1 = Roff ;
        Rch0 = Ron ;
    end ;
    I(IN0, OUTP) <+ V(IN0, OUTP) / transition(Rch0, 0, tr, tf) ;
    I(IN1, OUTP) <+ V(IN1, OUTP) / transition(Rch1, 0, tr, tf) ;
end
endmodule
```

```
//
// VerilogA for HinpLib, n_fet
//

`include "constants.h"
`include "discipline.h"

module n_fet(OUTP, AVSS, INP, SVSS);
output OUTP;
electrical OUTP;
inout AVSS, SVSS;
electrical AVSS, SVSS;
input INP;
electrical INP;

parameter   real Vth=2.5 from (0:5) ;
parameter   real Ron = 1k from (0:inf) ;
parameter   real Roff = 1e12 from (0:inf) ;
parameter   real tr=100p from [0:inf) ;
parameter   real tf=100p from [0:inf) ;

real Rch ;

analog begin
    @(initial_step) Rch=Roff ;
    if (analysis("static")) begin
        Rch = Roff ;
    end else
    begin
      if (V(INP) < Vth) Rch = Roff ;
      else Rch = Ron ;
```

```
    end ;
    I(OUTP, AVSS) <+ V(OUTP, AVSS) / transition(Rch,0, tr, tf) ;
end
endmodule
```

```
//
// VerilogA for HinpLib, n_follower, veriloga
//

`include "constants.h"
`include "discipline.h"

module n_follower(AVDD, AVSS, INP, OUTP, SVSS, VBN_SHAPER);
inout AVDD, AVSS, SVSS;
electrical AVDD, AVSS, SVSS;
inout INP, OUTP, VBN_SHAPER;
electrical INP, OUTP,VBN_SHAPER;
electrical      x ;

parameter  real  Rout = 1k ;
parameter  real  Vtn = 0.75 ;

analog begin

    V(x) <+ V(INP) - Vtn ;
    I(x, OUTP) <+ V(x, OUTP) / Rout ;

end
endmodule
```

```
//
// VerilogA for HinpLib, neg_pk_detect
//

`include "constants.h"
`include "discipline.h"

module neg_pk_detect(AVDD, AVSS, INP, OUTP, SVDD, SVSS, VB_SHAPER, rst,
track);
inout AVDD, AVSS, SVDD, SVSS;
electrical AVDD, AVSS, SVDD, SVSS;
inout INP, OUTP, VBN_SHAPER, rst,track;
electrical INP, OUTP, VBN_SHAPER, rst,track;

parameter    real   Vth = 2.5 from (1 : 4) ;
```

```
parameter    real   rst_level = 5.0 ;

real         outp_voltage ;

analog begin
     //
     // initialization stuff
     //
     @(initial_step) begin
          outp_voltage = rst_level ;
     end ;
     if (analysis("static")) begin
         outp_voltage = rst_level ;
     end
     else begin
       if (V(rst) > Vth) begin
            outp_voltage = rst_level ;
       end
       else
       if (V(INP) < outp_voltage) begin
          if (V(track) > Vth) outp_voltage = V(INP) ;
       end
      end ;
      V(OUTP) <+ outp_voltage ;
end
endmodule
```

```
//
// VerilogA for HinpLibMacro, nor_gate
//

`include "constants.h"
`include "discipline.h"

module nor_gate(Y, AVDD, AVSS, SVDD, SVSS, a0, a1);
output Y;
electrical Y;
inout AVDD, AVSS, SVDD, SVSS;
electrical AVDD, AVSS, SVDD, SVSS;
input a0, a1;
electrical a0, a1;

parameter real vlogic_high = 5;
parameter real vlogic_low = 0;
parameter real vtrans = 2.5;
```

```
parameter real tdel = 0 from [0:inf);
parameter real trise = 1p from (0:inf);
parameter real tfall = 1p from (0:inf);

real X;
integer logic1, logic2;

analog begin

    logic1 = V(a0) > vtrans;
    logic2 = V(a1) > vtrans;

    @ (cross(V(a0) - vtrans, 1))  logic1 = 1;
    @ (cross(V(a0) - vtrans, -1)) logic1 = 0;

    @ (cross(V(a1) - vtrans, 1))  logic2 = 1;
    @ (cross(V(a1) - vtrans, -1)) logic2 = 0;

    //
    // define the logic function.
    //

    X = (!(logic1 || logic2)) ? vlogic_high : vlogic_low;
    V(Y) <+ transition( X, tdel, trise, tfall);
end
endmodule
```

```
//
// VerilogA for HinpLib, nsw
//

`include "constants.h"
`include "discipline.h"

module nsw(OUTP, AVSS, SVSS, INP);
output OUTP;
electrical OUTP;
inout AVSS, SVSS;
electrical AVSS, SVSS;
input INP;
electrical INP;

parameter   real Vth=2.5 from (0:5) ;
parameter   real Ron = 1k from (0:inf) ;
parameter   real Roff = 1e12 from (0:inf) ;
```

```
parameter   real tr=100p from [0:inf) ;
parameter   real tf=100p from [0:inf) ;

real Rch, T ;

analog begin
    @(initial_step) Rch = Roff ;
     if (analysis("static")) begin
          Rch = Roff ;
     end else
     begin
       if (V(INP) > Vth) Rch = Ron ;
       else Rch = Roff ;
     end ;
     I(OUTP,AVSS) <+ V(OUTP,AVSS) / transition(Rch,0, tr, tf) ;
end
endmodule
```

```
//
// VerilogA for HinpLibMacro, one_shot_100ns
//

`include "constants.h"
`include "discipline.h"

module one_shot_100ns(AVDD, AVSS, INP, SVDD, SVSS, VBN_10uA, outp);
inout AVDD, AVSS, SVDD, SVSS;
electrical AVDD, AVSS, SVDD, SVSS;
inout INP, VBN_10uA, outp;
electrical INP, VBN_10uA, outp;

parameter real output_high = 5;
parameter real output_low = 0 ;
parameter real pulse_width = 100n ;
parameter real Vth = 2.5 ;

real   q_out ;

analog begin
      if (analysis("static")) begin
        @(initial_step) begin
             q_out = output_low ;
        end
      end
      if(V(INP) > Vth) q_out = output_high ;
```

```
    if (delay(V(INP), pulse_width) > Vth) q_out = output_low ;
    V(outp) <+ q_out ;
end
endmodule
```

```
//
// VerilogA for HinpLibMacro, one_shot_vari
//

`include "constants.h"
`include "discipline.h"

module one_shot_vari(AVDD, AVSS, INP, OUTP, SVDD, SVSS, vc);
inout AVDD, AVSS, SVDD, SVSS;
electrical AVDD, AVSS, SVDD, SVSS;
inout INP, OUTP, vc;
electrical INP, OUTP, vc;

parameter real output_high = 5;
parameter real output_low = 0 ;
parameter real Vth = 2.5 ;
parameter real tscale = 4u ;

real   q_out, pulse_width , tstop ;

analog begin
    if (analysis("static")) begin
      @(initial_step) begin
          q_out = output_high ;
          tstop = 9999 ;
      end
    end
    pulse_width = V(vc) * tscale ;
    if(V(INP) > Vth) begin
        q_out = output_low ;
        tstop = $realtime + pulse_width ;
    end ;
    if ($realtime >= tstop) begin
        q_out = output_high ;
        tstop = 9999 ;
    end ;
    V(OUTP) <+ q_out ;
end
endmodule
```

```
//
// VerilogA for HinpLib, or_3inp
//

`include "constants.h"
`include "discipline.h"

module or_3inp(Y, AVDD, AVSS, SVDD, SVSS, A0, A1, A2);
output Y;
electrical Y;
inout AVDD, AVSS, SVDD, SVSS;
electrical AVDD, AVSS, SVDD, SVSS;
input A0, A1, A2;
electrical A0, A1, A2;

parameter real vlogic_high = 5;
parameter real vlogic_low = 0;
parameter real vtrans = 2.5;
parameter real tdel = 0 from [0:inf);
parameter real trise = 1p from (0:inf);
parameter real tfall = 1p from (0:inf);

integer logic1, logic2,logic3;

real X;

analog begin
    logic1 = V(A0) > vtrans;
    logic2 = V(A1) > vtrans;
    logic3 = V(A2) > vtrans;
    @ (cross(V(A0) - vtrans, 1)) logic1 = 1;
    @ (cross(V(A0) - vtrans, -1)) logic1 = 0;
    @ (cross(V(A1) - vtrans, 1)) logic2 = 1;
    @ (cross(V(A1) - vtrans, -1)) logic2 = 0;
    @ (cross(V(A2) - vtrans, 1)) logic3 = 1;
    @ (cross(V(A2) - vtrans, -1)) logic3 = 0;

    //
    // define the logic function.
    //

    X = (logic1 || logic2 || logic3) ? vlogic_high : vlogic_low;
    V(Y) <+ transition( X, tdel, trise, tfall);
end
endmodule
```

```
//
// VerilogA for HinpLibMacro, or_gate
//

`include "constants.h"
`include "discipline.h"

module or_gate(Y, AVDD, AVSS, SVDD, SVSS, A0, A1);
output Y;
electrical Y;
inout AVDD, AVSS, SVDD, SVSS;
electrical AVDD, AVSS, SVDD, SVSS;
input A0, A1;
electrical A0, A1;

parameter real vlogic_high = 5;
parameter real vlogic_low = 0;
parameter real vtrans = 2.5;
parameter real tdel = 0 from [0:inf);
parameter real trise = 1p from (0:inf);
parameter real tfall = 1p from (0:inf);

integer logic1, logic2;

real X;

analog begin
    logic1 = V(A0) > vtrans;
    logic2 = V(A1) > vtrans;
    @ (cross(V(A0) - vtrans, 1)) logic1 = 1;
    @ (cross(V(A0) - vtrans, -1)) logic1 = 0;
    @ (cross(V(A1) - vtrans, 1)) logic2 = 1;
    @ (cross(V(A1) - vtrans, -1)) logic2 = 0;

    //
    // define the logic function.
    //

    X = (logic1 || logic2) ? vlogic_high : vlogic_low;
    V(Y) <+ transition( X, tdel, trise, tfall);
end
endmodule
```

```
//
// VerilogA for HinpLib, p_fet
//

`include "constants.h"
`include "discipline.h"

module p_fet(AVDD, AVSS, SVDD, node);
inout AVDD, AVSS, SVDD;
electrical AVDD, AVSS, SVDD;
inout node;
electrical node;

branch(AVDD,node)  res;
parameter  real  R=30K;

analog  begin
  I(res)  <+  V(res) / R ;
end
endmodule
```

```
//
// VerilogA for HinpLib, p_follower
//

`include "constants.h"
`include "discipline.h"

module p_follower(AVDD, AVSS, INP, OUTP, SVDD, SVSS, VBN_SHAPER);
inout AVDD, AVSS, SVDD, SVSS;
electrical AVDD, AVSS, SVDD, SVSS;
inout INP, OUTP, VBN_SHAPER;
electrical INP, OUTP, VBN_SHAPER;
electrical       x ;

parameter  real  Rout = 1k ;
parameter  real  Vtp = 0.95 ;

analog begin
    V(x) <+ V(INP) + Vtp ;
     I(x, OUTP) <+ V(x, OUTP) / Rout ;
end
endmodule
```

```
//
// VerilogA for HinpLib, pos_pk_detect
//

`include "constants.h"
`include "discipline.h"

module pos_pk_detect(AVDD, AVSS, INP, OUTP, SVDD, SVSS, VB_SHAPER, rst, track);
inout AVDD, AVSS, SVDD, SVSS;
electrical AVDD, AVSS, SVDD, SVSS;
inout INP, OUTP, VBN_SHAPER, rst, track;
electrical INP, OUTP, VBN_SHAPER, rst, track;

parameter    real   Vth = 2.5 from (1 : 4) ;
parameter    real   rst_level = 0.0 ;

real         outp_voltage ;

analog begin

     //
     // initialization stuff
     //

    @(initial_step) begin
          outp_voltage = rst_level ;
     end ;
     if (analysis("static")) begin
         outp_voltage = rst_level ;
     end
     else begin
       if (V(rst) > Vth) begin
            outp_voltage = rst_level ;
       end
       else
       if (V(INP) > outp_voltage) begin
              if (V(track) > Vth) outp_voltage = V(INP) ;
      end
     end ;
     V(OUTP) <+ outp_voltage ;
end
endmodule
```

```
//
// VerilogA for HinpLibMacro, pseudo_nor
//

`include "constants.h"
`include "discipline.h"

module pseudo_nor(OUT, AVDD, AVSS, SVDD, SVSS, A, B, C, E, d);
output OUT;
electrical OUT;
inout AVDD, AVSS, SVDD, SVSS;
electrical AVDD, AVSS, SVDD, SVSS;
input A, B, C, E, d;
electrical A, B, C, E, d;

parameter real vlogic_high = 5;
parameter real vlogic_low = 0;
parameter real vtrans = 2.5;
parameter real tdel = 0 from [0:inf);
parameter real trise = 1p from (0:inf);
parameter real tfall = 1p from (0:inf);

real X;
integer logic1, logic2,logic3,logic4,logic5;

analog begin
    logic1 = V(A) > vtrans;
    logic2 = V(B) > vtrans;
    logic3 = V(C) > vtrans;
    logic4 = V(d) > vtrans;
    logic5 = V(E) > vtrans;
    @ (cross(V(A) - vtrans, 1))  logic1 = 1;
    @ (cross(V(A) - vtrans, -1)) logic1 = 0;
    @ (cross(V(B) - vtrans, 1))  logic2 = 1;
    @ (cross(V(B) - vtrans, -1)) logic2 = 0;
    @ (cross(V(C) - vtrans, 1))  logic2 = 1;
    @ (cross(V(C) - vtrans, -1)) logic2 = 0;
    @ (cross(V(d) - vtrans, 1))  logic2 = 1;
    @ (cross(V(d) - vtrans, -1)) logic2 = 0;
    @ (cross(V(E) - vtrans, 1))  logic2 = 1;
    @ (cross(V(E) - vtrans, -1)) logic2 = 0;

    //
    // define the logic function.
    //
```

```
    X = (!(logic1 || logic2 || logic3 || logic4 || logic5)) ? vlogic_high : vlogic_low;
    V(OUT) <+ transition( X, tdel, trise, tfall);
end
endmodule
```

```
//
// VerilogA for HinpLibMacro, xor_gate
//

`include "constants.h"
`include "discipline.h"

module xor_gate(Y, AVDD, AVSS, SVDD, SVSS, A0, A1);
output Y;
electrical Y;
inout AVDD, AVSS, SVDD, SVSS;
electrical AVDD, AVSS, SVDD, SVSS;
input A0, A1;
electrical A0, A1;

parameter real vlogic_high = 5;
parameter real vlogic_low = 0;
parameter real vtrans = 2.5;
parameter real tdel = 0 from [0:inf);
parameter real trise = 1p from (0:inf);
parameter real tfall = 1p from (0:inf);

real X;

integer logic1, logic2;

analog begin
    logic1 = V(A0) > vtrans;
    logic2 = V(A1) > vtrans;
    @ (cross(V(A0) - vtrans, 1))  logic1 = 1;
    @ (cross(V(A0) - vtrans, -1)) logic1 = 0;
    @ (cross(V(A1) - vtrans, 1))  logic2 = 1;
    @ (cross(V(A1) - vtrans, -1)) logic2 = 0;

    //
    // define the logic function.
    //

    X = (logic1 ^ logic2) ? vlogic_high : vlogic_low;
    V(Y) <+ transition( X, tdel, trise, tfall);
```

```
end
endmodule
```

```
//
// VerilogA for HinpLib, zc
//

`include "constants.h"
`include "discipline.h"

module zc(agnd, AVDD, AVSS, INN, INP, OTA_AGND, OUTP, SVDD, SVSS,
          VBN_10uA, VBN_DISC);
inout agnd;
electrical agnd;
inout AVDD, AVSS, SVDD, SVSS;
electrical AVDD, AVSS, SVDD, SVSS;
inout INN, INP, OTA_AGND,OUTP, VBN_10uA, VBN_DISC;
electrical INN, INP, OTA_AGND,OUTP, VBN_10uA, VBN_DISC;

parameter real output_high = 5;
parameter real output_low = 0 ;
parameter real slope = 1G from (0:inf);

real    high_minus_low, high_plus_low ;

analog begin
    if (analysis("static")) begin
       @(initial_step) begin
         high_minus_low = 0.5*(output_high - output_low);
         high_plus_low  = 0.5*(output_high + output_low);
       end
     end
     V(OUTP) <+ high_minus_low * tanh(slope*V(INP, INN)) + high_plus_low;
end
endmodule
```

APPENDIX D

Testbench for the Macro Model Simulations

```
// Verilog test bench to simulate the macro model circuit.

`timescale 100ns/10ns
module tester1 (acq_all_L, acq_clk, common_stop,force_rst,force_track,
        global_cfd_en, id_in,quiet, reset, sclk, sin, token_into_chip,
        veto_rst_l);
   output acq_all_L,acq_clk ,common_stop, force_rst ,force_track ,global_cfd_en, quiet,
          reset, sclk, sin, token_into_chip, veto_rst_l;
   output [7:0] id_in;

   reg acq_all_L,acq_clk ,common_stop, force_rst ,force_track ,global_cfd_en, quiet,
          reset, sclk, sin, token_into_chip, veto_rst_l;
   reg [47:0] config_reg;
   reg [7:0] id_in;
   integer i;

      initial begin
         reset = 1'b1;
#1       reset = 1'b0;
      end
      initial begin

// Disable aff CFDs

      global_cfd_en = 1'b0;

// Set chip id to #0

      id_in = 8'b00000000;

// Make token into chip active

      token_into_chip = 1'b0;

// Do not assert the acq_all signal

      acq_all_L = 1'b1;

// Hold acq_clk low

      acq_clk = 1'b0;
```

```
// Dont veto the reset

      veto_rst_l = 1'b1;

//Deassert common_stop and FORCE A RESET

      common_stop = 1'b0;
      force_rst = 1'b0;
      force_track = 1'b0;
      quiet = 1'b0;

// Enable the CFDs

#1000 ;
#260 force_track = 1'b1;
#30   force_rst = 1'b1;
#1    force_rst = 1'b0;
#1    global_cfd_en = 1'b1;

// Assert common stop at t = 130.5u

#13   common_stop = 1'b1;

// Assert veto reset at t = 132.2u

#17   veto_rst_l = 1'b0;
#27   global_cfd_en = 1'b0;

// Ok to release the common stop signal

#1    common_stop = 1'b0;

//Aquire the results at 136u

#10   for(i=0;i<=16;i=i+1)
          begin
          acq_clk= 1'b1;
          #1;
          acq_clk = 1'b0;
          #1;
          end
       force_track = 1'b0;
#30   veto_rst_l = 1'b1;
#1066 force_rst = 1'b1;
#1   force_rst = 1'b0;
#1   global_cfd_en = 1'b1;
```

```
#13  common_stop = 1'b1;
#17  veto_rst_l = 1'b0;
#27  global_cfd_en = 1'b0;
#1    common_stop = 1'b0;
// Acquisition Clock running at 10MHz

#10   for(i=0;i<=16;i=i+1)
          begin
          acq_clk= 1'b1;
          #1;
          acq_clk = 1'b0;
          #1;
          end
#1   global_cfd_en = 1'b1;
#30  force_rst = 1'b1;
#1 force_rst = 1'b0;
#1 veto_rst_l = 1'b1;
end

 initial begin

// Loading the configuration register.....
// Take 4.8u when a clock 10MHz is used
// Start at 200ns

// To enable all CFDs place 0s

     config_reg[31:0]= 32'b00000000000000000000000000000000 ;

// Bit 32 is neg_pulse (0=> negative pulses at the CSA output)

     config_reg[32] = 1'b1;

// Bit 33 is one_usec_l (0=>1 usec range selected for TVC )

     config_reg[33] = 1'b0;

// Bit 34 is lo_gain (0=>high gain mode selected )

     config_reg[34] = 1'b0;

// Bit 35 is test_mode (1=>test mode selected )

     config_reg[35] = 1'b1;

// Bit 36 is test_mode2 (1=>test mode selected )
```

```
    config_reg[36] = 1'b0;

// Bit 37 is test_mode3 (1=>test mode selected )

    config_reg[37] = 1'b0;

// Bit 38-39 currently unused

    config_reg[39:38] = 2'b00;

// Set chip id

    config_reg[47:40] = 8'b00000000;

#2  for(i=47;i>=0;i=i-1)
    begin
        #1 sclk = 1'b0;
        sin = config_reg[i];
        #1 sclk = 1'b1;
     end
end
endmodule
```

```
//
// Verilog code for Programming the DACs
//

`timescale 100ns/10ns

module program_dacs (dac_stb, data_in, sel_ext_addr);
output dac_stb, sel_ext_addr;
output [5:0] data_in;

reg dac_stb, sel_ext_addr, dac_sign ;
reg [5:0] data_in;

integer channel_number, dac_value;

initial begin
    channel_number = 0;
    dac_value = 0;
    dac_sign = 1'b0;
    data_in = 6'b000000;
    dac_stb = 1'b0;
```

```
// We dont want external adddressing for the time being
// Dont program the DACs now

     sel_ext_addr = 1'b0;

// Wait for the configuration register to be loaded
// Wait for 12 us
#1000 ;
#120  sel_ext_addr = 1'b1;
        data_in[4:0] = int2v(channel_number);
        data_in[5] = 1'b0;
// Put data on data_in bus
// Address gets latched on the rising edge of the dac_stb

#1     dac_stb = 1'b1;
#1     data_in[4:0] = int2v(dac_value);
        data_in[5] = dac_sign;

// Dac data gets latched on the falling edge of the dac_stb

#1     dac_stb = 1'b0;
#1     sel_ext_addr = 1'b0;

end

// Function to convert a integer to a vector

function [4:0]  int2v ;
input a;
integer a,i;
begin
     for(i=0;i<5;i=i+1)
     begin
       if(a % 2)
          int2v[i] = 1'b1;
       else
          int2v[i] = 1'b0;
     a = a/2;
     end
end
endfunction
endmodule
```

APPENDIX E

Ocean Scripts to Automate Simulations

```
;
; Ocean script to obtain the transfer characteristics of a circuit.
;

simulator('spectre)
design( "/home/mentor/cds/Hinp/simulation/bandgap/spectre/schematic/netlist/netlist")
resultsDir( "/home/mentor/cds/Hinp/simulation/bandgap/spectre/schematic/")
modelFile("/home/cdsadmin/vendors/AMI/mods/typ")
temp(27)
desVar("tdet" 10m)
analysis('tran ?start 0 ?stop 13m ?maxstep 500n)
run()
selectResults('tran)
ocnPrint( v("VREF") ?output "/home/mentor/cds/Hinp/excel.dat/bandgap_10m_1.txt"
?numberNotation 'none)
ocnPrint( v("AVDD") ?output "/home/mentor/cds/Hinp/excel.dat/bandgap_10m_2.txt"
?numberNotation 'none)
;close(fid)

;
; Ocean script to find the temperature dependence of a circuit
;

simulator( 'spectre )
design("/home/mentor/cds/Hinp/simulation/bandgap/spectre/schematic/netlist/netlist")
resultsDir( "/home/mentor/cds/Hinp/simulation/bandgap/spectre/schematic" )
modelFile(("/home/cdsadmin/vendors/AMI/mods/typ"))

fid = outfile("/home/mentor/cds/Hinp/excel.dat/bandgap_temp.txt" "w")
val =-40
for( i 0 12
desVar( "tdet" 100u)
temp(val)
analysis('tran ?stop "5m" ?outputstart "3m" ?maxstep "100n" )
save( 'all)
run()
openResults("/home/mentor/cds/Hinp/simulation/bandgap/spectre/schematic/psf")
selectResult( 'tran)
outp_v = ymax(v( "/VREF"))
outp_i = ymax(i( "/I2/M30/S")) *1e6
fprintf(fid "%d %1.9f %f\n" val outp_v outp_i )
val=val+10
```

```
)
close(fid)


;
; Ocean script to determine the walk of a circuit
;

simulator( 'spectre )
design("/home/mentor/cds/Hinp/simulation/cfd_walk/spectre/schematic/netlist/netlist")
resultsDir( "/home/mentor/cds/Hinp/simulation/cfd_walk/spectre/schematic" )
modelFile('("/home/cdsadmin/vendors/AMI/mods/typ"))
temp(27)

fid = outfile("/home/mentor/cds/Hinp/excel.dat/cfd_walk.txt" "w")

val =2.675
tmp =6.5m
for( i 0 50
  desVar( "vdet" val)
  analysis('tran ?stop "600n" ?maxstep "100p" )
  run()
  openResults("/home/mentor/cds/Hinp/simulation/cfd_walk/spectre/schematic/psf")
  selectResult( 'tran)
  half_point= cross( v("/cfd_out") 2.5 1 'rising)
  walk = (half_point - 0.5u) * 1e9
  amplitude = (ymax( v("/input")) - 1.5)
  fprintf(fid "%f %f \n" amplitude walk)
  val=val + tmp
)
close(fid)

;
; Ocean script to determine the rise time and the linearity of a circuit
;


simulator( 'spectre )
design("/home/mentor/cds/Hinp/simulation/csa_tran/spectre/schematic/netlist/netlist")
resultsDir( "/home/mentor/cds/Hinp/simulation/csa_tran/spectre/schematic" )
modelFile('("/home/cdsadmin/vendors/AMI/mods/typ"))
temp(27)
fid = outfile("/home/mentor/cds/Hinp/excel.dat/csa_lin_ln.txt" "w")
val =0.055625u
for( i 0 24
desVar( "idet" val)
```

```
analysis('tran ?stop "200u" ?outputstart "140u" ?maxstep "20n" )
run()
openResults("/home/mentor/cds/Hinp/simulation/csa_tran/spectre/schematic/psf")
selectResult( 'tran)
max_v = (ymax( v("/CSA_OUT"))-ymin( v("/CSA_OUT")))
electrons = (val*80e-12)/1.602e-19
Mev = electrons * 3.6e-6
rt = riseTime( v("/CSA_OUT") 150u t 151u t 10 90 )* 1e9
fprintf(fid "%f %1.9f %f\n" Mev max_v rt )
val=val * 2.15443469
)
close(fid)

;
; Ocean script to perform noise analysis
;

simulator( 'spectre )
design( "/home/mentor/cds/Hinp/simulation/csa_noise/spectre/schematic/netlist/netlist")
resultsDir( "/home/mentor/cds/Hinp/simulation/csa_noise/spectre/schematic/")
modelFile('("/home/cdsadmin/vendors/AMI/mods/typ"))
temp(27)
const_cap =2.5p
fid = outfile("/home/mentor/cds/Hinp/excel.dat/csa_noise_newhg.txt" "w")
val = 0
for(i 1 15
val = i * 10p
desVar("cdet" val)
analysis( 'noise ?start "1" ?stop "10M" ?p "OUT" ?n 0 ?iprobe "V3")
run()
v = rmsNoise( 10k 1M )
electrons =( (v * const_cap )/1.602e-19)
capvalue = val * 10e11
fprintf(fid "%4.0f %4.0f \n" electrons capvalue )
)

close(fid)

;
; Ocean script to find the peaking time and the amplitude of a circuit
;

simulator('spectre)
design( "/home/mentor/cds/Hinp/simulation/csa_shaper_tran/spectre/config/netlist/netlist")
resultsDir( "/home/mentor/cds/Hinp/simulation/csa_shaper_tran/spectre/config/")
modelFile("/home/cdsadmin/vendors/AMI/mods/typ")
```

```
temp(27)
fid = outfile("/home/mentor/cds/Hinp/excel.dat/shaper_npeaktime.txt" "w")
val =1.5
for( i 0 16
desVar( "idet" 10m)
desVar( "vdet" val)
analysis('tran ?stop "75u" ?outputstart "49u" ?maxstep "20n" )
run()
openResults("/home/mentor/cds/Hinp/simulation/csa_shaper_tran/spectre/config/psf")
selectResult( 'tran)
peaktime = (xmin( v("/OUTP") 1)-50u)*1e6
peakvalue = (ymax( v("/OUTP"))-ymin( v("/OUTP")))*1e3
fprintf(fid "%f %f %f \n" peaktime peakvalue val)
val = val + 0.1
)
close(fid)
```

APPENDIX F

PROBE PADS

Name	Description
1. out_full_tvc	-- Output of the TVC in channel 0.
2. TVC_OUT	-- Output of the TVC (Input to source follower).
3. vb_tvc	-- Bias voltage for TVC.
4. tvc_cap_gnd	--TVC cap ground voltage.
5. ana_rst	-- Analog reset (resets the TVC).
6. CFD_OUT	-- CFD output that goes to the pad.
7. out_cfd	-- CFD output connected to input of the switch in channel 0.
8. veto_rst_L	-- Veto reset (active low) that is connected to the pad.
9. force_rst	-- Force reset that is connected to the pad.
10. dig_rst	-- Digital reset that is connected to the pad.
11. common_stop	--Common stop (tvc stop) that is connected to the pad.
12. DLY_VC	--DLY_VC voltage.
13. acq_ack	--acquisition acknowledge pull down node.
14. vbp_hit	--VBP_HIT voltage.
15. MULT	--MULT which is the input signal buffer in the common channel.
16. ORgate	--ORgate pull down nodes in the hit logic circuits.
17. encode	--encode signal from hit logic circuit in channel 0.
18. acq_all_L	--acq_all_L signal that is connected to the pad.
19. acq_clk	--acquisition clock that is connected to the pad.
20. token_in	--token_in signal into channel 0.
21. Hit	--Hit signal (output of FF1) in the hit logic circuit in channel 0.
22. vbn_10uA	--VBN_10uA voltage
23. le_out	--Output of LE in channel 0.
24. en_cfd	--enable CFD signal for channel 0.
25. zc_out	-- Output of ZC in channel 0.
26. neg-pulse	--neg_pulse (configuration register bit 32).
27. DVDD	--Digital VDD.
28. vbn_disc	--VBN_DISC voltage.
29. DGND	--Digital ground.
30. inp (LE)	--Input signal to LE in channel 0.
31. dac_out	--Output of DAC in channel 0.
32. AGND----- (DAC)	--AGND (close to DAC).
33. SUBSTRATE	--Substrate voltage.
34. vbp1_dac	--VBP1_DAC voltage.
35. dac_stb (qual_dac_stb)	--Qualify DAC strobe signal from common digital circuit.
36. dac(4)	--Input to DAC(bit 4)
37. dac(5)	--Input to DAC(bit 5)
38. dac(3)	--Input to DAC(bit 3)
39. dac(2)	--Input to DAC(bit 2)
40. dac(1)	--Input to DAC(bit 1)

41. dac(0) --Input to DAC(bit 0)
42. ZC_INM --ZC inp- in channel 0.
43. ZC_INP --ZC inp+ in channel 0.
44. Quiet --Quiet signal connected to the pad.
45. chan_sel --channel select (from decoder) of channel 0.
46. PEAK_OUT --Output of peak sampler connected to source followers.
47. out_sample_peak -- Output of peak sampler (input to switch) in channel 0.
48. vbn_shaper --VBN_SHAPER voltage.
49. force_track -- force track signal connected to the pad.
50. gain_amp_out --Output of gain amplifier in channel 0.
51. peak_pfoll --Output of positive peak follower in channel 0.
52. peak_nfoll -- Output of negative peak follower in channel 0.
53. SHAPER_OUT -- Output of shaper (output of the switch) connected to pad.
54. test_mode_3 --Test_mode_3 signal(Configuration register bit 37)
55. out_shaper -- Output of shaper(input to switch) in channel 0
56. EXT_SHAPER --External input to shaper connected to the pad.
57. PEAK_TIME --Peak Time (V_C) voltage.
58. AVSS --Analog ground.
59. AVDD --Analog VDD.
60. out_and1 --Signal that controls SW1 and SW2 in channel 0.
61. test_mode_2 -- Test_mode_2 signal(Configuration register bit 36)
62. out_csa -- output of CSA (input to switch) in channel 0.
63. CSA_OUT -- Output of CSA (output of the switch) connected to pad.
64. test_mode -- Test_mode signal (Configuration register bit 35)
65. OUTP-nowlin --OUTP signal of nowlin (Input to ZC+).
66. OUT-nowlin --OUT signal of nowlin (Input to LE).
67. AGND---nowlin --AGND (close to nowlin circuit).
68. OUTN-nowlin --OUTN signal of nowlin (Input to ZC-).
69. lo_gain --Lo gain signal (Configuration register bit 34)
70. vbn_csa --VBN_CSA voltage.
71. CSA_AVSS --CSA ground.
72. CSA_AVDD --CSA VDD.
73. CSA_GND --CSA_GND voltage.

Probe Pads in Common channel

74. Equal --Equal signal in common digital circuit.
75. Input to BJT1 -- Emitter of BJT Q1
76. Input to BJT2 -- Emitter of BJT Q2
77. VREF -- VREF voltage in the band gap.
78. Input to BJT3 -- Emitter of BJT Q3