KLOPF—A Classically Conditioned Silicon Neuron Architecture Specification CSE 529 - 1988

Rev 3.0

Kenneth M. Weigel
Jon W.T. Inouye

Technical Report CS/E-88-033
20 September 1988

Oregon Graduate Center
19600 S.W. von Neumann Drive
Beaverton, Oregon 97006-1999

KLOPF -- A Classically Conditioned Silicon Neuron
Architecture Specification
CSE 529 - 1988

Rev 3.0

Kenneth M. Weigel
Jon W. T. Inouye

August 3, 1988

1. Abstract

The KLOPF chip will be a two input - one output silicon neuron that will be able to learn three stage classical conditioning, delay conditioning, backward conditioning, and allow for experiments on strength of input versus acquisition time. More complex behavior can be obtained from this architecture using more inputs, but for the scope of this project it is felt that it is better to have a subset of the full model that functions rather than attempt a larger circuit that we may not be able to complete or test. Each chip will contain two functionally equivalent neurons. By connecting chips together in a linear fashion, an N-input single output neuron may be formed without requiring any external logic.

Each neuron consists of a digital input register connected to a pair of digital weights. One weight represents the excitatory connection and the other weight represents the inhibitory connection. Each weight has been implemented as a binary up/down counter. Digital to analog circuitry interface the inputs/weights to the analog circuits implementing the learning algorithm. The counters are incremented and/or decremented according to a dynamic voltage level within the analog circuitry. The output is in analog form. (Therefore multi-layer networks will require external circuitry.)

The learning algorithm requires a single unconditional stimulus, and at least one conditional stimulus. The major difference between the two is that the unconditional stimulus has fixed weights while the conditional stimulus has connections which should be allowed to change during the learning process. To differentiate between the conditioned and unconditioned stimulus, there are control pins which fix the values in a weight, i.e. disable weight changes. Additionally, this feature allows the digital weights to be fixed and read out for examination. The chip(s) may share a single data/address bus for this purpose. Weights are adjusted according to changes in input (presynaptic) and output (postsynaptic) activity, weight strength, and a learning rate constant. Note that a change in input activity will be remembered for a period of time after it occurs.

A basic learning session consists of providing the conditional stimulus followed by the unconditional stimulus. Note that no learning should occur should both stimuli be presented simultaneously. In the case of classical conditioning, the chip should learn to respond to the conditional stimulus. The chip will be implemented in a 3-micron 2-metal CMOS process using both analog and digital techniques.

2. Background

This chip is based on the drive reinforcement neuron model developed by Harry Klopf at the Air Force Wright Patterson Aeronautical Laboratories [3]. The form of two input, single output drive reinforcement model can be seen in figure 1. Each input is connected to a summation unit via a pair of weights. One weight represents the excitatory connection while the other represents the inhibitory connection.

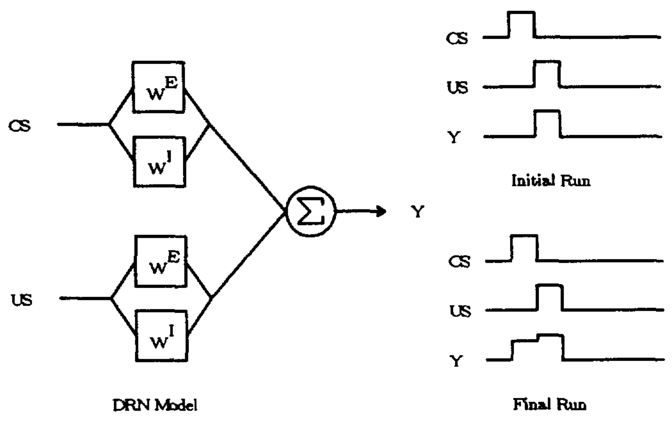


Figure 1 DRN Model with Desired Response

The output of the chip can be described by the following equation

$$y(t) = \sum_{i=1}^{n}(w_i^E(t) - w_i^I(t))x_i(t)$$

where $y(t)$ is the output signal, $x_i(t)$ is the input signal, $w_i^E(t)$ is the excitatory weight connected from the i^{th} input signal to the response mechanism, which does the summation in a traditional neural net fashion. The inhibitory connection is represented by $w_i^I(t)$.

The chip implements Klopf's learning algorithm by adjusting the plastic weights connecting the conditional stimulus to the response mechanism. In Klopf, the weights are changed according to the following equation

$$\Delta w_i(t) = \Delta y(t)\sum_{j=1}^{\tau} c_j |w_i(t-j)| \Delta x_i(t-j)$$

where $\Delta w_i(t) = w_i(t+1) - w_i(t)$, $\Delta y(t) = y(t) - y(t-1)$, and $\Delta x_i(t-j) = x_i(t-j) - x_i(t-j-1)$. The number of time steps for which the input will be remembered is denoted by τ and c_j represents how strongly previous signals will be remebered. We assume things that happen most recently will be remebered better than things occuring futher in the past, i.e. $c_j > c_{j+1}$. The weights can only be changed by a positive increase in signal activity, i.e. Δx_i must be positive in order to change a weight. All negative Δx_i values should be ignored.

Since Δx_i and c_j should always be positive and w_i is represented by an integer, the direction of the change is dependent on $\Delta y(t)$. Since our model contains only positive weights, we had to modify the formula when updating our inhibitory weights. The formula applies equally to both weights, except Δw_i is added to the excitatory weight and subtracted from the inhibitory weight. Therefore, a positive increase in Δy results in an increase in the excitatory weight and a decrease in the inhibitory weight (should Δw_i be non-zero).

Note that this method requires memory in order to store the previous values of the input signals and their weights. The weights connected to the unconditioned stimulus should not be changed by this algorithm. In this model, weights are not allowed to go to zero; any weight which does so stays at zero using this learning algorithm. In our implementation, we are replacing the time sampled signals with real-time analog signals which eliminates the memory requirements. A decaying exponential function replaces the learning rate constants. Departures from the Klopf time sampled method will be noted in the modules it occurs in.

Figure 2 shows the difference between our real-time analog decay function compared to a digital time-step implementation. The digital implementation has the advantage of flexibility since different c_j values can be used. The advantages of the analog implementation is space complexity. The analog circuitry does not require any register memory nor adders.

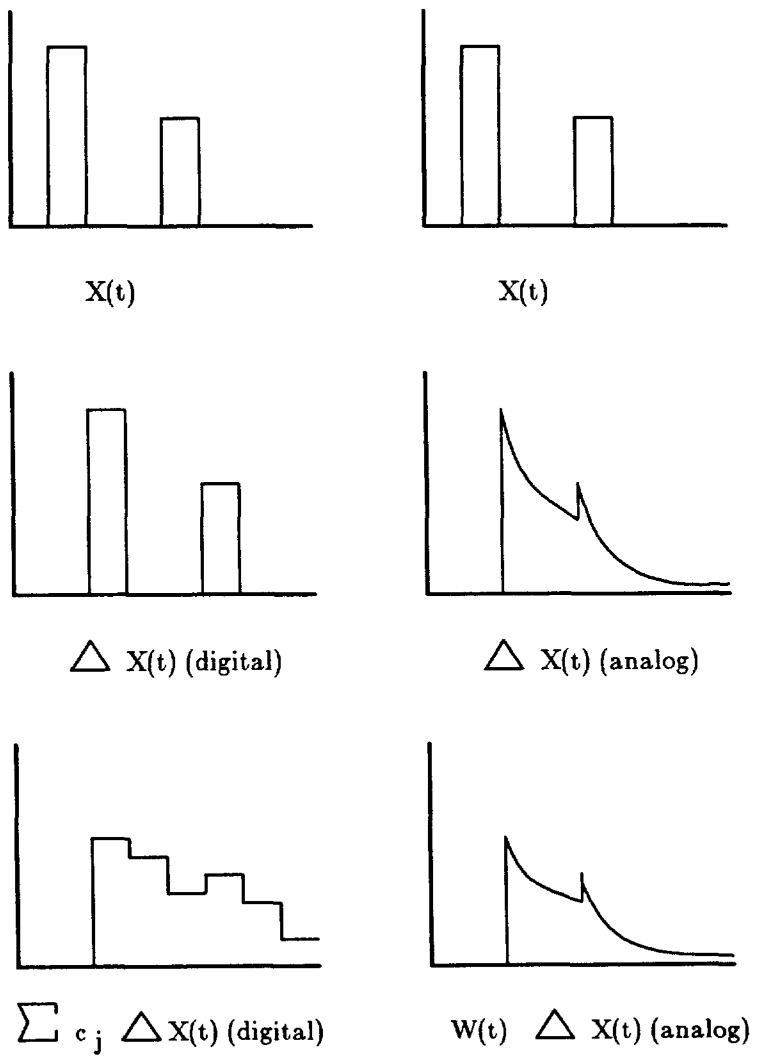


Figure 2 Desired Response Curves

3. Microarchitecture

The KLOPF chip contains two functionally equivalent "neurons". Each neuron has been built using six modules. The functional block partition can be seen in figure 3.

The input module (INMOD) contains the interface between the digital section and the analog circuitry. INMOD contains the input-registers and the weight-counters plus a small amount of circuitry to handle the I/O pads.

CONVMOD, the conversion module, is made up of the D/A converters (DAC's) which transform the input and weight values into analog signals. CONVMOD also contains the Δx circuit which calculates the change in the input, and delays it for a small (but significant!) amount of time.

The output module (OUTMOD) generates the output, Y. OUTMOD also calculates the change in Y, ΔY, and the correct count direction based on ΔY. Note that ΔY determines whether a weight should incremented or decremented.

The calculation of the weight change, ΔW, is performed by DWMOD. DWMOD has a leaky integrator, which does the summation over time of the input signal, and the c_j calculation from Klopf's equation. DWMOD then mutiplies this value against the ΔY signal to obtain a current level signifying the amount of change in its weight.

The voltage-to-frequency module (VFMOD) transforms the current into a pulsed waveform. VFMOD takes a bipolar input (for an up or down count) and converts it to pulses (count signals) sent to the digital weights.

BIASMOD is very small; it takes an external reference current, and mirrors from it the necessary bias currents and voltages for the full circuit.

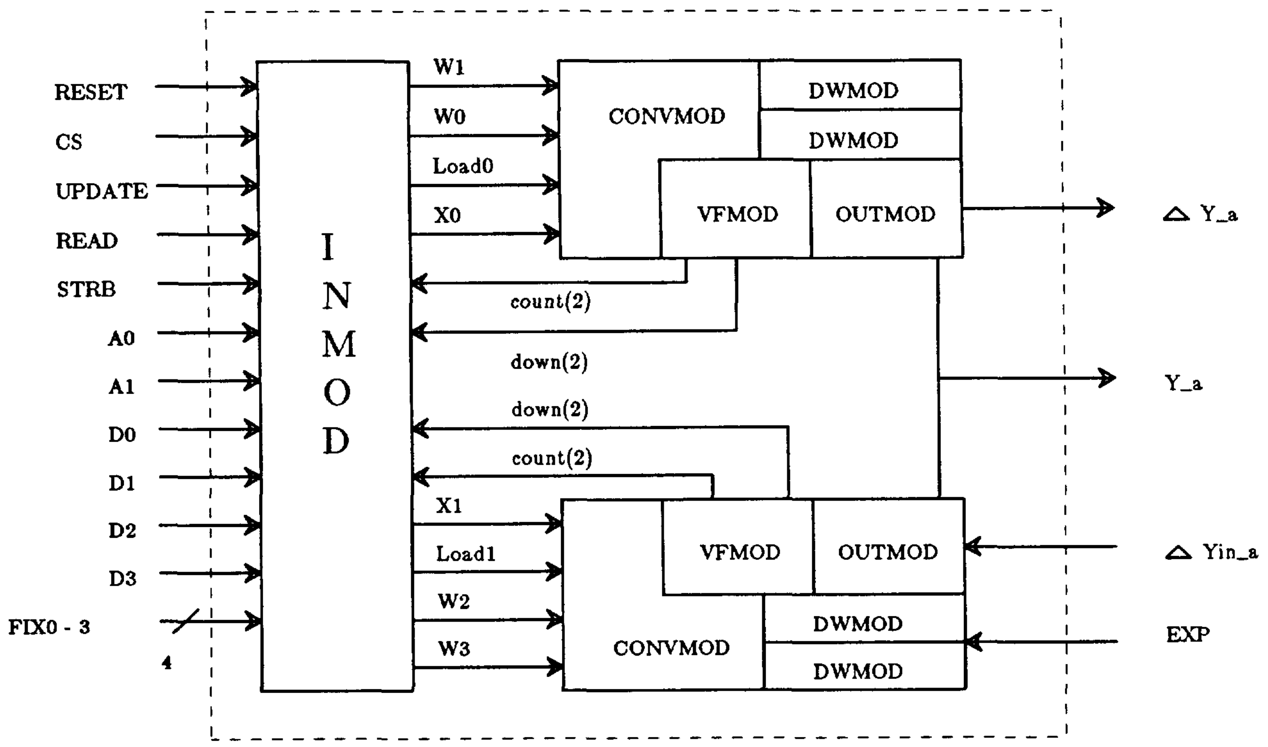


Figure 3 Functional Block Partition

We have designed the output module to allow it to cascade from other chips to form an N-input Klopf neuron. This is done by bringing out two pins Y_0 and Y_1 and an EXPAND from the input to the summer in OUTMOD. By taking the Y signal from many chips and connecting them to the EXPAND pin, a Y signal that is the sum of all inputs is generated. The Y signal that is chosen for cascading is then connected to all the Y_inX pins and they will all generate the same deltaY. The ΔW module (DWMOD) takes $\Delta x_i w_i$ value and integrates the value with a leaky integrator. This approximates the Klopf c_j behavior and eliminates any kind of data storage of discrete timesteps in either the analog or digital domain. There is one DWMOD for each weight. The output of each leaky integrator is connected to the input of a current output analog multiplier which has ΔY for its other output. This sum is fed into a V/F converter (VFMOD) which then converts the input current to some frequency. VFMOD responds to bipolar currents, so the multiplier output current needs no extra signal conditioning and a comparator will look at the sign of ΔY to generate an UP/DOWN signal.

INPUTS:

A0, A1 -- Address bus, used to select the registers and counters
D0 to D3 -- Data bus
RESET -- Asynchronous reset
CS -- Chip select
UPDATE -- Enables loading of inputs. If not asserted enables loading of weights
STRB -- Strobe signal, loads selected registers with contents on data bus
FIX0 to FIX3 -- Assertion of these lines disable counters (from counting)
EXPAND -- Analog input used for expansion to an N input neuron
Yin_0,Yin_1 -- Analog inputs to the ΔY circuit

OUTPUTS:

Y_a -- Analog output (response)
ΔY_a -- Analog change in output

3.1. Input Module (INMOD)

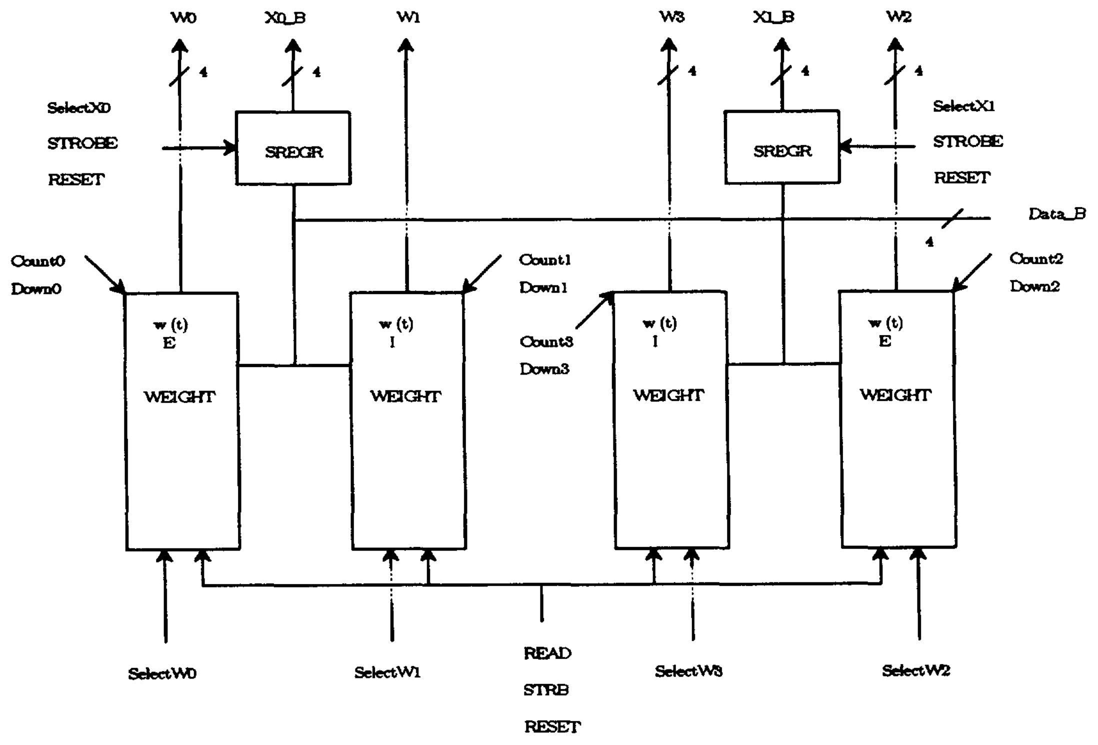


Figure 4 INMOD Functional Partition

The input module receives the input signals (both weight and stimulus values) in digital form and stores them in registers. The weight values are stored in a counter register that allow an initial value to be put into the counters which then can be modified by the learning rule. INMOD is shown in figure 4.

The plastic weights are stored in special counters which can be either incremented or decremented by pulsed signals from the VFMOD circuit. The counters will have special logic such that they will neither overflow, nor be decremented to 0.

The registers will consist of static latches connected to the data bus by transmission gates. The counters will be implemented using a combination of static latches and full adders.

INPUTS:

RESET -- asyncronous reset (from pins)
CS, UPDATE, Address_B -- Used to select correct registers (from pins)
Data_B -- Data bus (from pins)
Down0-3 -- Indicates the direction in which to count (from VFMOD)
Count0-3 -- Indicates an increment or decrement should occur (from VFMOD)

OUTPUTS:

X0 -- Input register 0 (to CONVMOD)
X1 -- Input register 1 (to CONVMOD)
Load_0 -- Signals new input in X0 (to CONVMOD)
Load_1 -- Signals new input in X1 (to CONVMOD)
W0 -- Excitatory weight of input 0 (to CONVMOD)
W1 -- Inhibitory weight of input 0 (to CONVMOD)
W2 -- Excitatory weight of input 1 (to CONVMOD)
W3 -- Inhibitory weight of input 1 (to CONVMOD)

3.2. Conversion Module (CONVMOD)

CONVMOD consists of five digital to analog converters (DACS) and a special delta circuit for differentiation and time delay. CONVMOD is responsible to converting all digital signals (one input and a pair of weights) into analog signals used by OUTMOD and DWMOD. CONVMOD also produces a delayed, decaying signal representing the change in the input.

INPUTS:

X -- Digital 4-bit input value (from INMOD)
We -- Digital 4-bit excitatory weight value (from INMOD)
Wi -- Digital 4-bit inhibitory weight value (from INMOD)
Load_n -- Signal indicating new input value (from INMOD)

OUTPUTS:

X_a -- Analog value of input (to OUTMOD)
We_a -- Analog value of excitatory weight (to OUTMOD)
Wi_a -- Analog value of inhibitory weight (to OUTMOD)
XWe_a -- Analog product past change in X and excitatory weight (to DWMOD)
XW1_a -- Analog product past change in X and inhibitory weight (to DWMOD)

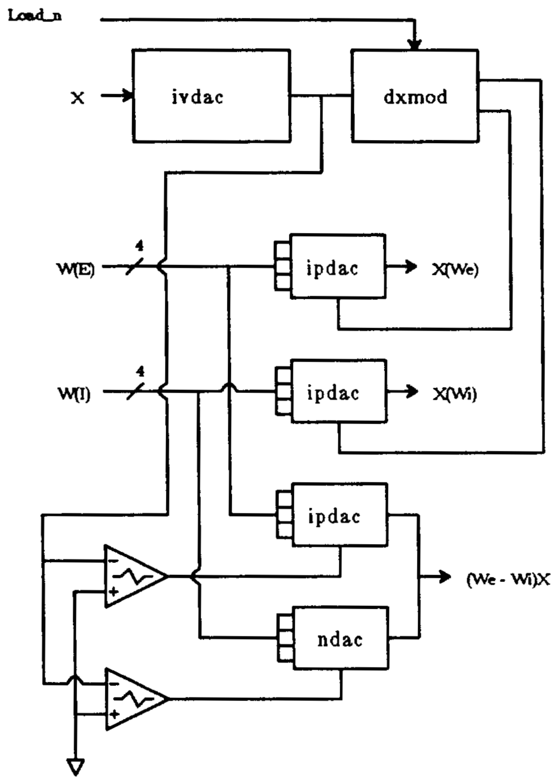


Figure 5 CONVMOD Functional Partition

3.3. Output Module (OUTMOD)

The output module (OUTMOD) receives the products $w_i(t)x_i(t)$ and calculates the output y where and Δy is the derivative of y. Inhibitory inputs were inverted to put them in the proper phase for summing. The output module includes an input EXPAND which will allow the formation of an N input Klopf neuron. All Y amplifiers will be in parallel, so the dY circuits can be connected to any of the Y_out pins.

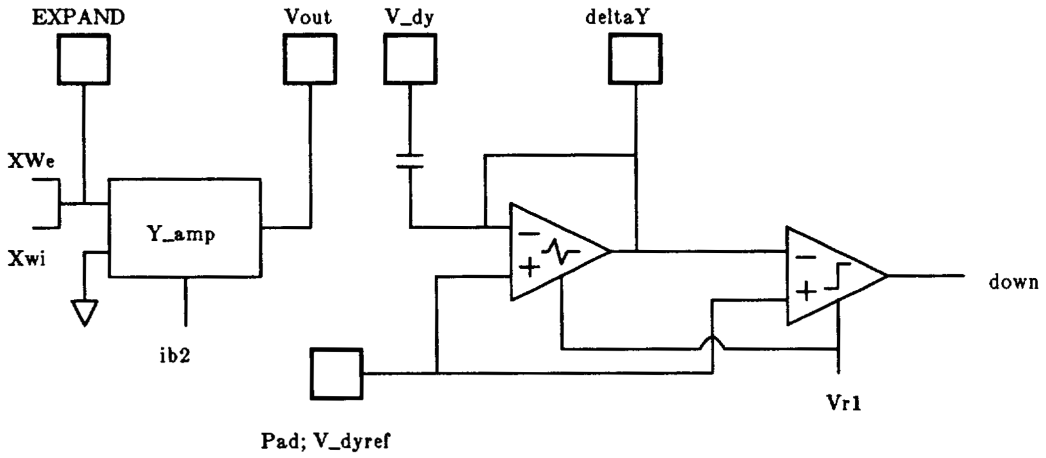


Figure 6 OUTMOD Functional Partition

INPUTS:

XOWe_a -- Product of X0 and excitatory weight
XOWi_a -- Product of X0 and inhibitory weight
X1We_a -- Product of X1 and excitatory weight
X1Wi_a -- Product of X1 and inhibitory weight
EXPAND_a -- Partial sum of products from another neuron

OUTPUTS:

Y_a -- Summed output in analog form
ΔYin_a -- Change in output response in analog form

3.4. Delta W Module (DWMOD)

The delta W module (DWMOD) performs analog integration on the signal $\Delta x_i w_i$ to replace the discrete step summation which is used in the Klopf simulator. While this step takes away the flexibility in modifying c_j it simplifies the circuit and eliminates the analog shift registers. The output of the integration is multiplied with ΔY to form the change in the weight. The direction of the change is determined by a comparator whose input is ΔYin. The result is converted by the VFMOD to pulses for the weight counters.

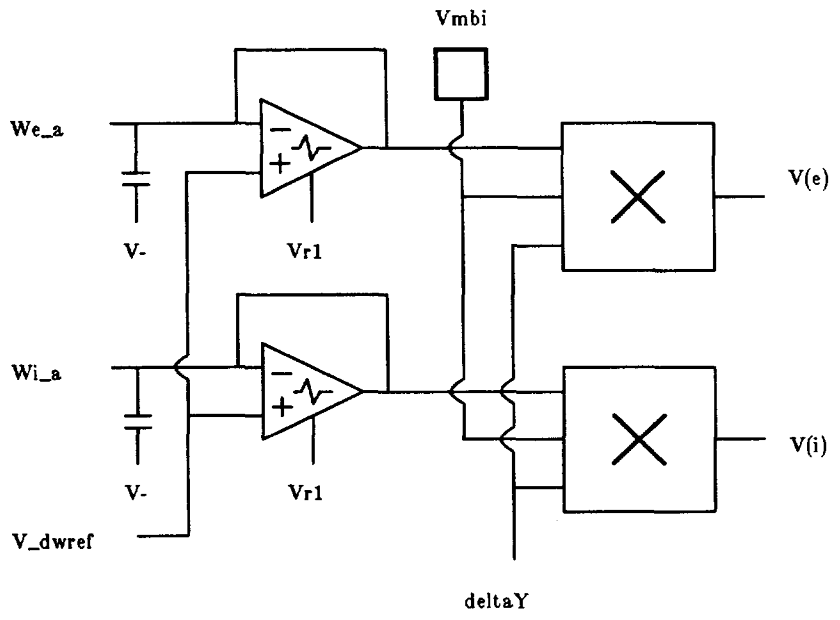


Figure 7 DWMOD Functional Partition

INPUTS:

ΔX0We_a -- Analog product (from INMOD)
ΔX0Wi_a -- Analog product (from INMOD)
ΔX1We_a -- Analog product (from INMOD)
ΔX1Wi_a -- Analog product (from INMOD)
deltaYin_a -- Change in response (from PINS)

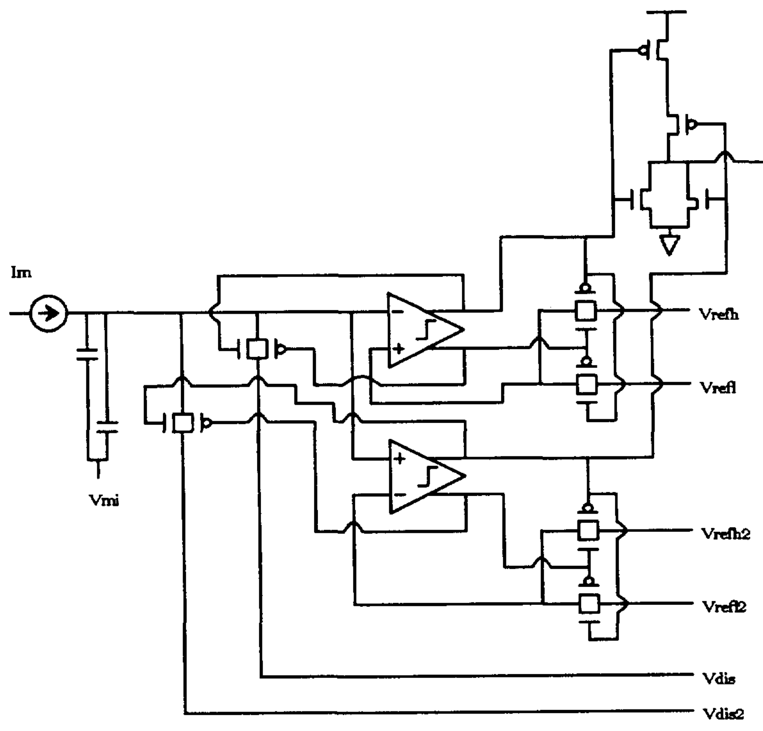
OUTPUTS:

V0_a -- Change in excitatory weight of X0 (to VFMOD)
V1_a -- Change in inhibitory weight of X0 (to VFMOD)
V2_a -- Change in excitatory weight of X1 (to VFMOD)

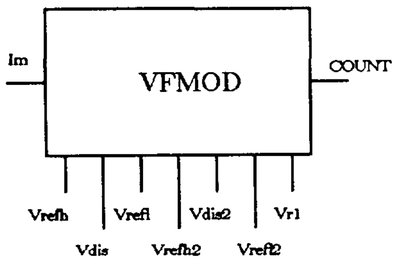
V3_a -- Change in inhibitory weight of X1 (to VFMOD)

3.5. V/F Module (VFMOD)

VFMOD is a very simple V/F converter that will convert the output of DWMOD to pulses. In the schematic, Im is the input current; it is the current from any one of the multipliers. The output from the multipliers is bipolar; VFMOD must be able to respond properly to that. To achieve this, there are actually two separate V/F converters working on the input current in parallel. For example, assume the voltage on the capacitor reaches Vrefh; the top comparator will trip, causing the capacitor voltage to discharge to Vdis. The second input of the comparator is switched to Vrefl to assure that the capacitor is substantially discharged, providing some hysteresis. If this feature were not in, under some conditions the V/F would only dischagrge the capacitor a few tens of millivolts causing severe error in the count. The same sort of thing happens with the lower comparator except the signs are reversed. The outputs from the two comparators are combined by the NOR gate to get a single COUNT signal, and the sign information (count up or down) is taken from the delta_Y circuit.



Schematic



Symbol

Figure 8 VFMOD Functional Partition

INPUTS:

I0_a -- Analog current output from multiplier 0
I1_a -- Analog current output from multiplier 1
I2_a -- Analog current output from multiplier 2
I3_a -- Analog current output from multiplier 3
Vrefh-- Input voltage, typically 1V
Vrefl-- Input voltage, typically 100mV
Vdis -- Input voltage, typically ground
Vrefh-- Input voltage, typically -1V

Vrefl2-- Input voltage, typically -100mV
Vdis2 -- Input voltage typically ground

OUTPUTS:

Count0-3 -- Signals a change in the weight

3.6. Bias Module (BIASMOD)

BIASMOD generates many of the reference currents and some of the reference voltages needed internally. It was decided to use some external references to allow for experimentation on the effect of varying some of the voltages on the learning rate, particularly in VFMOD. Ir1 is five times the reference voltage in from the pad; it and Ir2 go to the two amplifiers in each channel. Ir3 goes to the X input dac in the module IVDAC; this provides a full scale current equal to that of the TCAMP's. Vr1 is equal to a minimum geometry Vt at 2uA; this provides the non-critical bias to the TCAMP's and comparator modules.

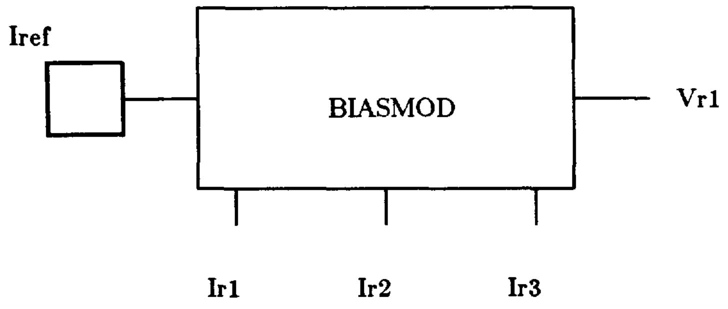


Figure 9 BIASMOD Functional Partition

INPUTS:

Iin -- Input current

OUTPUTS:

Vr1 -- Reference voltage 1 (for general bias use)
Ir1 -- 10uA reference to Y_amp amplifier
Ir2 -- 10uA reference to V_dy amplifier
Ir3 -- 2uA reference to dac in IVDAC

4. External Input/Output Specification

The digital section of the chip requires a 4-bit data bus, 2-bit address bus, and nine control pins. Both inputs and weights will be loaded into to chip using the data bus. Weights may also be read via the data bus. There are a number of analog pins, mostly inputs which will be used. A few pins will be used to help test the analog sections. An example would be to study the effect of reduced bias current and some modified reference levels on learning time.

The design requires three pad type, I/O pads (PadIO), input pads (PadIn), and analog pads (PadAnalog).

I/O Pins Specification		
Name	Function	# Pins
Data	Data Bus	4

Analog Pins Specification		
Name	Function	# Pins
EXPAND	Cascaded input (W). Analog product.	1
Vmi_a Vpl_a AGND	Analog power supplies	3
Vdwref	reference voltage	1
Vtiref	reference voltage	1
Vdis Vrefh Vrefl	VFC+ reference voltages	3
Vdisi Vrefhi Vrefli	VFC- reference voltages	3
Iref	Input reference current. All bias currents scale with respect to this.	1
Vmbi	Multiplier reference voltage	1
Y_out1 Y_out0	Output of Y. Response.	2
dY_0 dY_1	Input to ΔY generators	2
V_dxref	Reference for dxmod.	1

The Y, ΔYin, and EXPAND signals will be used to form an N-input 1-output Klopf neuron which may be able to show some higher conditioning abilities. EXPAND is the partial sum of products ($x_i w_i$) from other chips. ΔY is sent by the prime neuron and is accepted as ΔYin in secondary neuron chips. This allows secondary neurons to evaluate their new weights.

Input Pins Specification		
Name	Function	# Pins
Address	Address Bus	2
RESET	Sets all weights to minimum value, all inputs to zero, and discharges all analog levels.	1
CS	Chip select. Enables inputs and weights to be loaded. Also allows weights to be read.	1
UPDATE	Indicates inputs should loaded rather the weigths.	1
STRB	Loads the values presented on the data bus, or presents the values within the chip onto the data bus. (Depends on READ)	1
READ	Allows plastic weights to be read.	1
FIX0 to FIX3	Disables the counters, i.e. prevents the counters from changing state.	4

Note that the weights can be loaded separately. The FIX signals must be asserted while registers are being loaded. The STRB then determines the rate at which input data enters the chip, but has no effect on the internal analog update time. Therefore, the frequency of the STRB signal is confined to a certain range.

Chip Pins Specifications					
Pin #	Pad #	Name	Pin #	Pad #	Name
1	35	Blank	21	24	dY_0
2	36	EXPAND	22	23	dY_1
3	37	V_dxref	23	22	Y_out1
4	38	Iref	24	21	FIX2
5	39	Blank	25	20	FIX3
6	19	VPL_a	26	0	GND
7	18	Vrefl	27	1	RESET
8	17	Vrefh	28	2	CS
9	16	Vdis	29	3	A0
10	15	Blank	30	4	A1
11	14	Blank	31	5	UPDATE
12	13	Vtiref	32	6	READ
13	12	Vdwref	33	7	FIX1
14	11	Vmbi	34	8	FIX0
15	10	VML_a	35	9	Vdd
16	29	AGND	36	30	STRB
17	28	Vrefli	37	31	Data0
18	27	Vrefhi	38	32	Data1
19	26	Vdisi	39	33	Data2
20	25	Y_out0	40	34	Data3

The table above identifies the placement of our pins. The pin number corresponds to the pin chart given to us, while the pad numbers correspond to the pad numbers given on our frame. See Appendix B for the listing of pin placements for our pad frame.

5. External Interface

The following section describes how to use the KLOPF chip.

5.1. Initial State

The KLOPF chip needs to be run in a strict manner for it to function correctly. Upon power up the chip should be reset (RESET pin held high). This resets all analog sums to zero, all inputs to zero, and resets the weights to 1. This is the default initial state. Chip select (CS) should be deasserted and all FIX lines should be asserted.

5.2. Reading Weights

The weights can be read at any point in time. This ability will be used for debugging and conditioning analysis. Note that all weights (counters) should be fixed by asserting all FIX signals. This prevents the weight values from changing while they are being read. Data bus signals are valid while STRB is asserted.

```
              Initial Reset                        Reading Weights

               ________________________            ________________________
CS           /                          \        /                          \

                                                       _____
STROBE       __________________________          _____/     \______________

               ________________________            ________________________
FIXx         /                          \        /                          \

UPDATE       XXXXXXXXXXXXXXXXXXXXXXXXXX          __________________________

               ________________________            ________________________
READ         /                          \        /                          \

               ____
RESET        /     \___________________          __________________________

A BUS        XXXXXXXXXXXXXXXXXXXXXXXXXX          ===================XXXXXX

D BUS        XXXXXXXXXXXXXXXXXXXXXXXXXX          XXXXX=====XXXXXXXXXXXXXXXX
```

5.3. Loading Initial Weights

Since all counters reset to 1, the values for the excitatory and inhibitory weight of the unconditional stimulus must be loaded into the counters. CS must be high and READ and UPDATE must be low. A1 and A0 determine which weight should be selected.

5.4. Loading Inputs

The difference between loading inputs and loading weights is that UPDATE must be high. A1 determines which input should be selected.

```
          Loading Weights                      Loading Inputs (registers)

          _________________________            _________________________
CS        /                         \          /                         \
               _____                                _____
STROBE    ____/     \______________            ____/     \______________
          _________________________            _________________________
FIXx      /                         \          /                         \
                                               _________________________
UPDATE    _________________________            /                         \

READ      \_________________________/          \_________________________/

RESET     __________________________           __________________________

A BUS     ===============XXXXXXXXXXX           ===============XXXXXXXXXXX

D BUS     ===============XXXXXXXXXXX           ===============XXXXXXXXXXX
```

5.5. Learning Process

The learning process consists of running a number of trials for a given learning pattern. Note that the inputs must be loaded, even though they do not change in value. This alerts the analog circuitry that the next time step is occuring. CS should be low when weight changes are occuring. Don't forget to enable the weights (counters) once the inputs have been loaded!

Running a Trial

```
          __________________
CS        /                  \____________
           ____      ____
STROBE    /    \____/    \_______________
          __________________
FIXx      /                  \____________
          __________________
UPDATE    /                  \____________
                             _____________
READ      \_________________/

RESET     _________________________________     X - Unstable signal

A BUS     ======XXX==========XXXXXXXXXXXX        = - Stable signal

D BUS     ======XXX==========XXXXXXXXXXXX
```

6. Cell Descriptions

This section describes the structure and functions of the MAGIC cells used to implement the modules. Cells will be described in alphabetical order.

6.1. amplifier.mag

The amplifier is shown in figure 10; it is a complex circuit, and the reader is referred to the paper by Ahuja [1]. It was chosen for its relativley small size and good capacitive load driving capability.

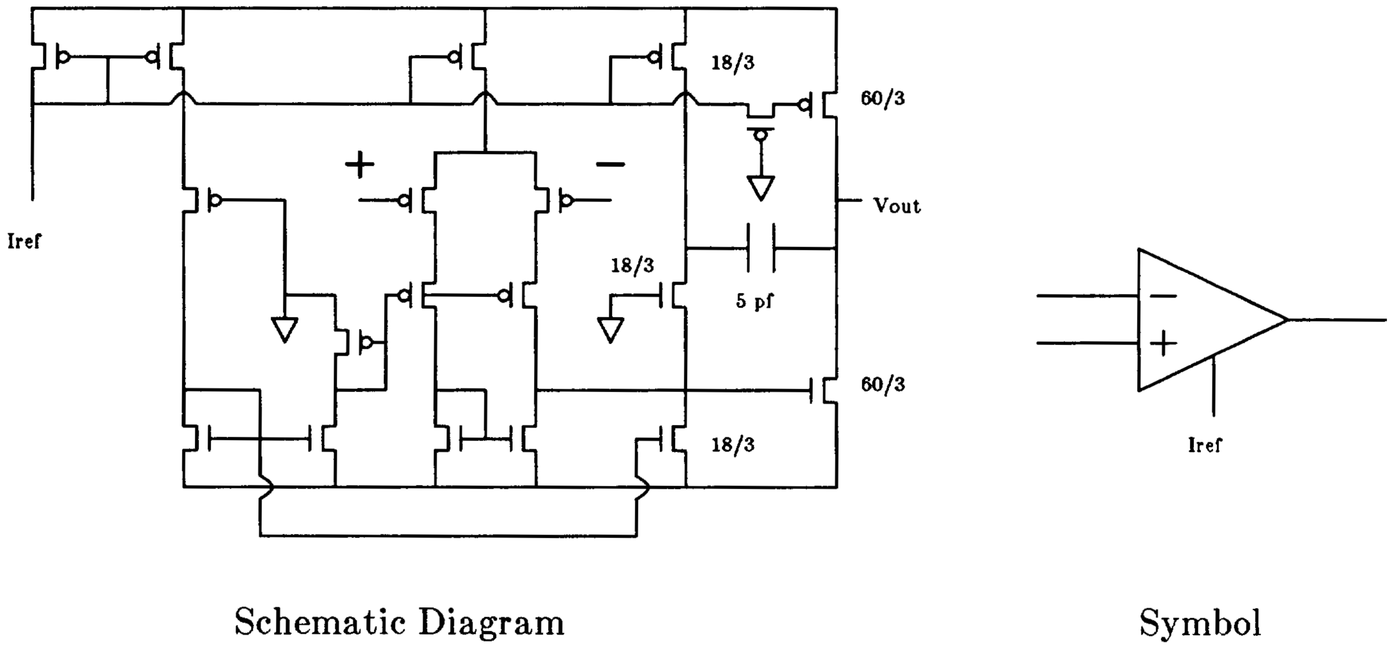


Figure 10 Amplifier

6.2. biasgen.mag

The bias circuit is shown in figure 11. It is extremely simple; a voltage, VR1 is generated with an external voltage to be the threshold voltage of a transistor at the current input. This biases up most of the circuit and is not critical. The amplifier needs about 10uA to function; since the nominal reference current is 2uA, these devices are 5x as large as the others. The IPDAC in the IVDAC block needs a bias current about equal to the bias of the TCAMP's; Ir3 provides that.

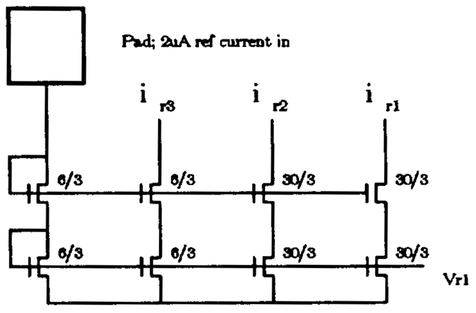


Figure 11 BIAS Circuit

6.3. budcntr0.mag

Counter cells have been implemented using static latches, full adders, and static transmission gates. The schmatic diagram of a counter cell (BUDCNTR0.MAG) is shown in figure 12.

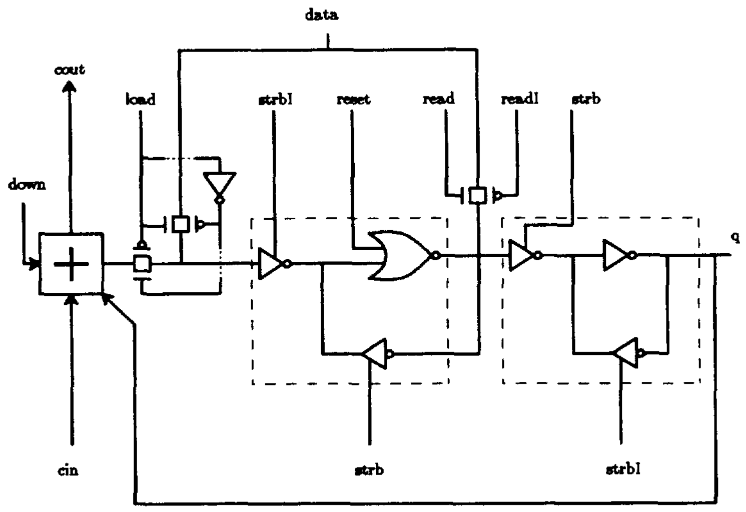


Figure 12 BUDCNTR0 - Binary Up/Down Counter Cell with RESET

6.4. budcntr1.mag

This cell is identical to BUDCNTR0.MAG except for a few minor alterations. BUDCNTR1.MAG uses SDLATCHS.MAG rather than SDLATCHR.MAG and includes in inverter to obtain a poisitive set value. As indicated, this cell resets to 1 rather than 0.

6.5. clogic.mag

This cell contains the decoding logic to form the select lines. See netlist for more description.

6.6. compl.mag

The comparator is a fairly straightforward design with two gain stages and both OUT and OUT_I available. It uses the reference voltage Vrl to bias the circuit to its nominal state. The schematic is shown in figure 13.

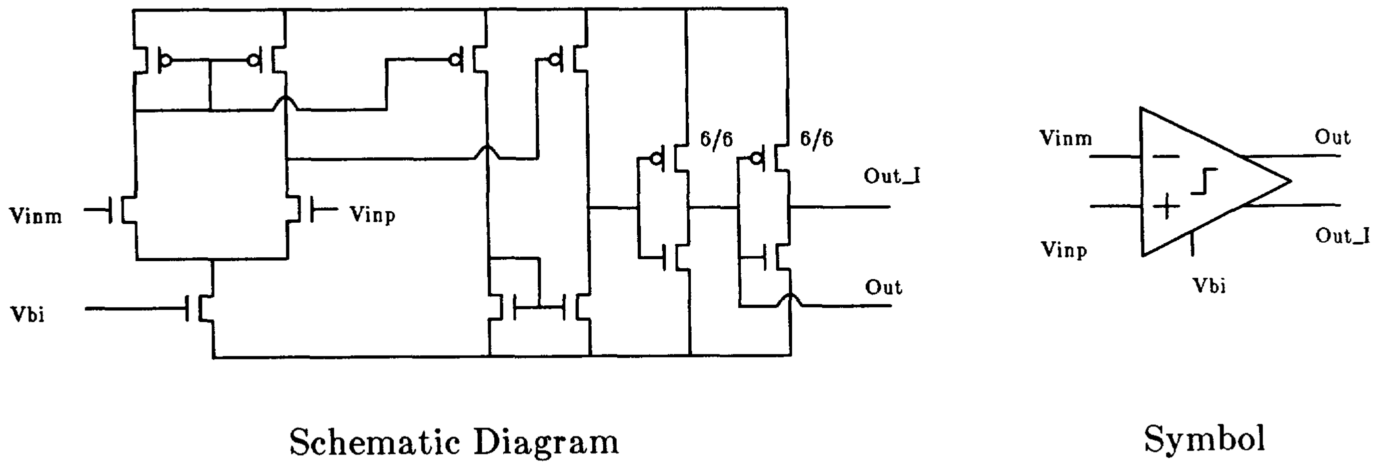


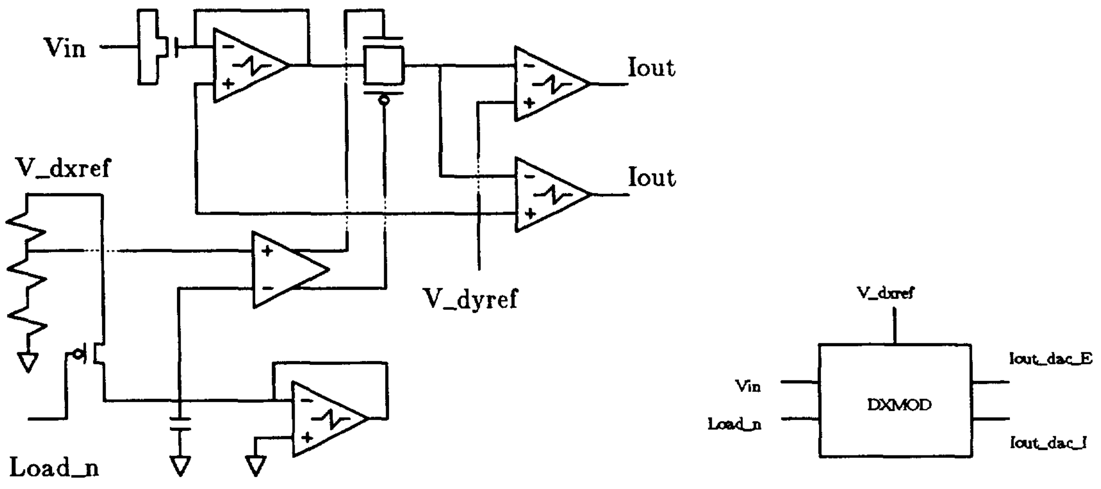
Figure 13 Comparator

6.7. dwmod.mag

DWMOD is a simple block, and the full schematic is shown in the section on block diagrams. Refer back to that for full information.

6.8. dxmod.mag

DXMOD is the second most complex of the analog modules ; the Klopf learning algorithm requires that no dx signal gets through coincidental with the input signal. It is the function of dxmod to generate dx from X, and to delay it. The dx part iof the circuit uses a V_dy module. To delay it, when the X dac is loaded, it charges a cap which discharges exponentially; this is compared to 1/3 of the reference voltage, which is about the time for one time constant (.30* reference voltage) before it opens the transmission gate to allow the signal through. The output signal then goes to TCAMPs wired as V-I convertord to feed into dwmod.



Schematic Diagram

Symbol

Figure 14 Delta X Circuit

6.9. faddr.mag

Standard full adder made from nand and xor gates.

6.10. fullkl.mag

The full circuit is shown in a block diagram form in figure 15. It is the way all the previously discussed blocks and modules are wired together.

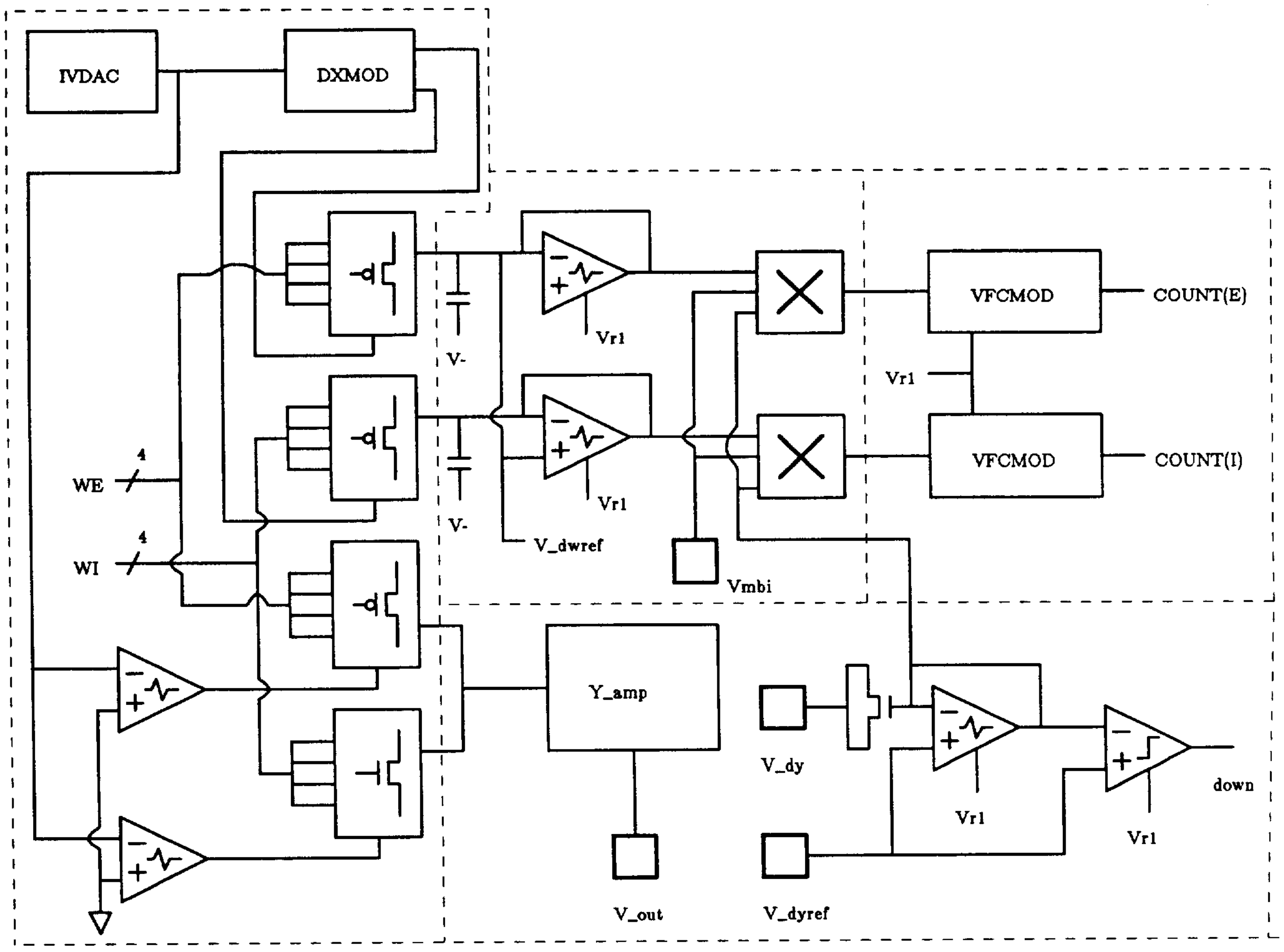
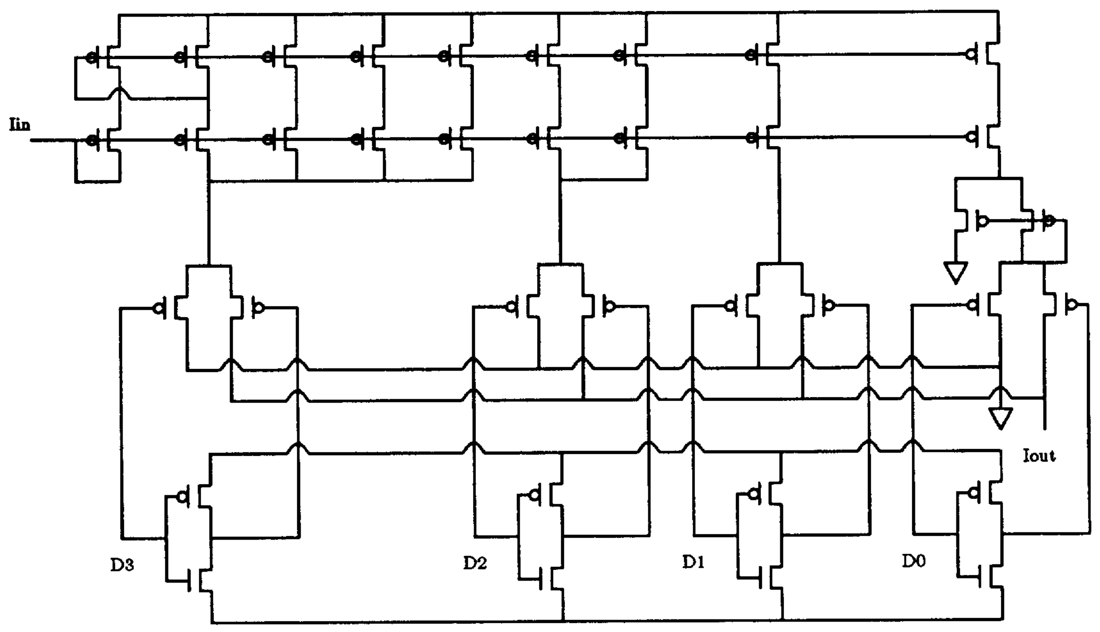
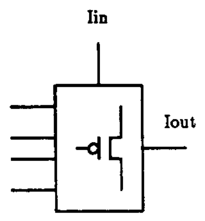


Figure 15 Analog Circuit Block Diagram

6.11. ipdac.mag

The IPDAC is simply a dac made up of P-transistors. The I in front refers to a minor change made to it; the module name has stuck. This module is very similar to the APN DAC; note the change made to the LSB circuitry to allow for some layout savings. The LSB current flows through a differential pair which roughly splits the current flowing in it; in this manner, all the subsequent bits can be half the size they otherwise would be. The current steering switches were not scaled, either; simulation showed this to be unnecessary in this application. Finally, the other departure from the APN DACs is that the APN DACs had gain of about 7.5; that is, full scale output current could be 7.5 times the input current. In our design it will only be 7.5/8 of the input current, or slightly less than unity. IPDAC is shown in figure 16.



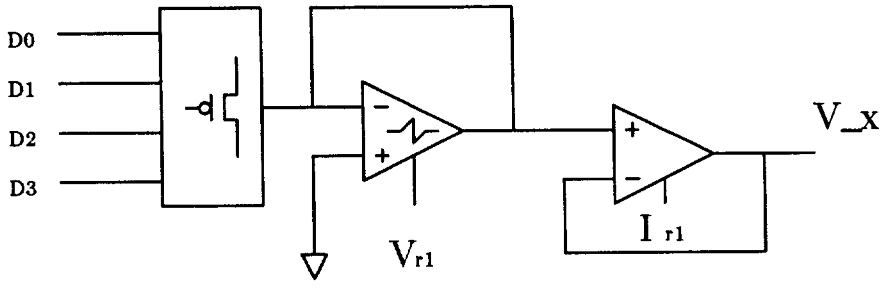


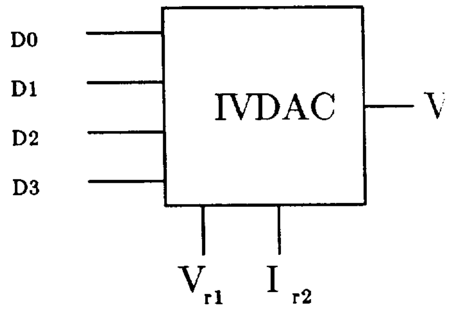
Schematic Diagram Symbol

Figure 16 IPDAC

6.12. ivdac.mag

The IVDAC is a Y_amp attached to an IPDAC module (figure 17). It provides the X_a voltage that is fed to dxmod, and a couple of TCAMP's are wired as V-I converters to power the excitatory and inhibitory DACs. Full scale voltage is about 1V.





Schematic Diagram Symbol

Figure 17 IVDAC

6.13. m1m2polycap.mag

Capacitor of 5pf.

6.14. mult2.mag

The analog multipliers use the $I_d=K(V_{gs}-V_{th})$ first order law of transistor physics for a transitor in saturation [2]. The circuit shown in figure 18 uses that characteristic to form the equation $I_{out}=K(V2-V_{th})(V1P-V1)$ where K is a constant. Note that this circuit only works for $V1A\&V1B>V_{th}$ and $V2>max(V1,V1P)$ (since the transistors must be in saturation). V_{th} is the threshold voltage with reference to V_{mi}.

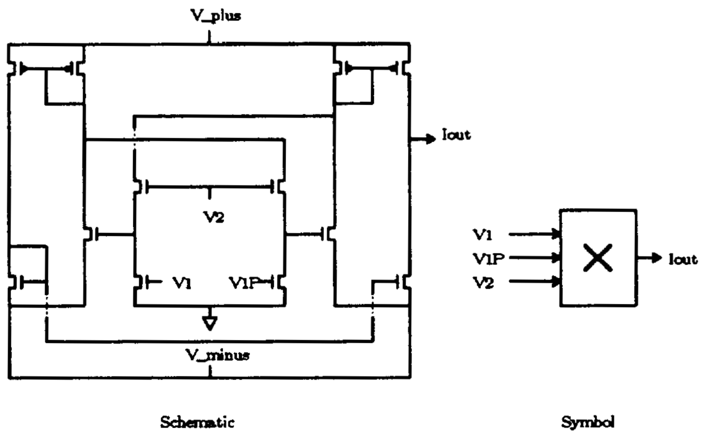


Figure 18 Two-quadrant Analog Multipliers

6.15. ndac.mag

NDAC is simply an N transistor version of the PDAC, shown in figure 19. All comments that were made for IPDAC apply to it.

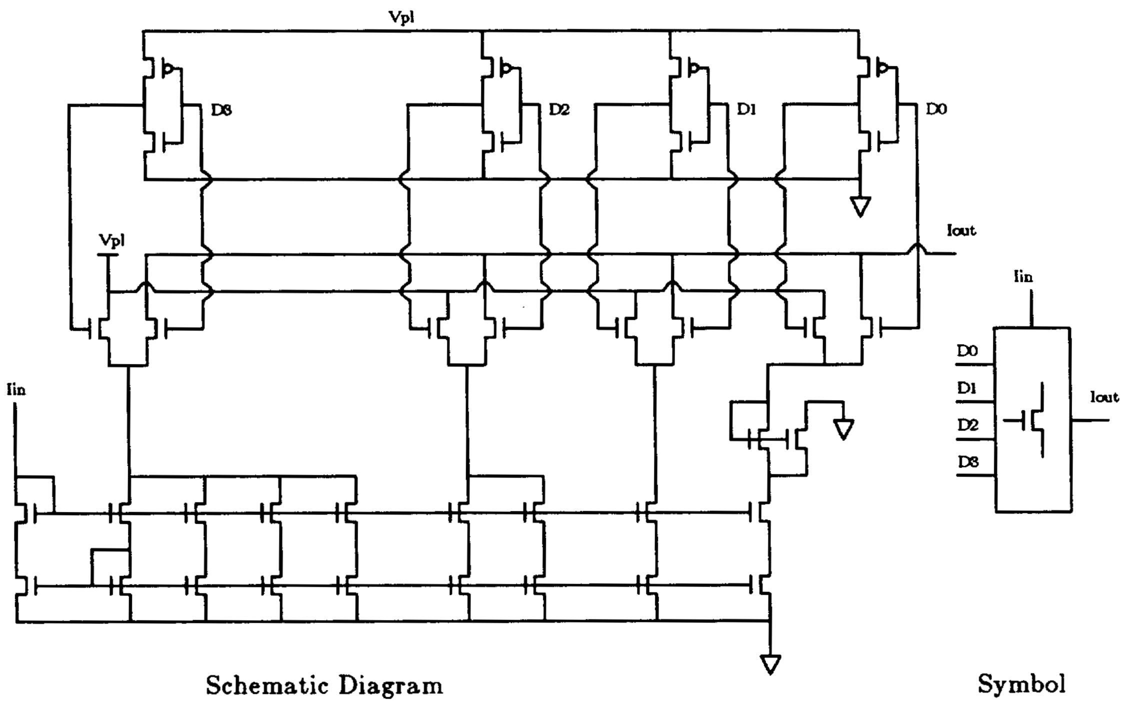


Figure 19 NDAC

6.16. polyndiffcap.mag

Another capacitor, using poly and the n diffusion.

6.17. resref.mag

Resistor divider reference voltage derived from the power supply, made from a long polysilicon line with various reference points. Used by dxmod. (See figure 14.)

6.18. sdlatchs.mag

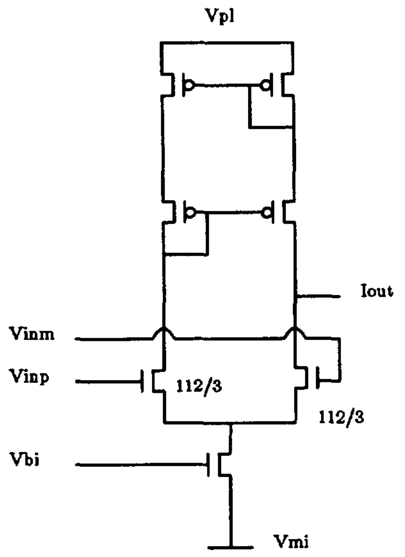
Static D-latch with SET asserted low. See netlist for more details.

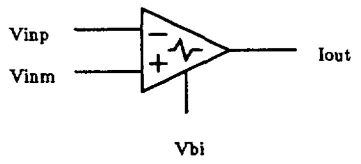
6.19. sregr.mag

The input registers where simply arrays of static register cells (SDLATCHR.MAG) with some combinational logic for loading values from the data bus.

6.20. tcamp.mag

The TCAMP is a transconductance amplifier, shown in figure 20, along with the symbol used for it in the block diagram. The TCAMP is a simple differential pair with a Wilson mirror; Vbi which sets up the bias current comes from BIASMOD. the only distinguishing feature of this block is that M1 and M2 are very long, about 80 lambda in the layout. this gives a reasonable input voltage range to about 1V. The TCAMP is very versatile in spite of its simplicity. As shown in the diagram it is a V-I converter. Ground VINP, tie VINM to Iout, and it behaves like a resistor when its input is driven. This is where the resistors come from for all the time constants. TCAMP was based on the simple transcunductance amplifier described by Mead [4].





Schematic Symbol

Figure 20 Transconductance Amplifier

6.21. vfc3.mag

The VFC is the most complex analog module. It responds to bipolar input currents and required a lot of SPICE finessing before it would work, and is shown in figure 8. The circuit functiion has been described previously; vfc3.mag provides the circuitry associated with the top comparator and switches; it also contributes one capacitor at the input.

6.22. vfc3i.mag

vfc3i.mag is the circuitry associated with the lower comparaqtor in figure 8, and is simply vfc3.mag with the phasing of the inputs and switches reversed. this aalso contributes one capacitor to the input.

6.23. V_dy.mag

V_dy is shown in figure 21; it uses a TCAMP as a resistor and takes the derivative of the input signal. The FET capacitor must have a couple volts across it to maintain a stable capacitance. Therefore, V_dyref is about 3 volts above ground. When there is a change in Vin, the voltage a the input to the TCAMP changes by the same amount; then the TCAMP will discharge it to whatever V_dyref is.

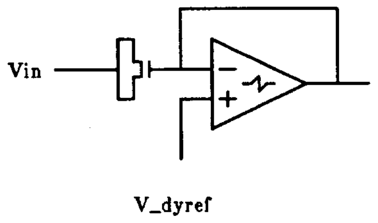


Figure 21 V_dy

6.24. weight.mag

The weights will be implemented using binary up/down counters. The counters should not overflow, neither should their value fall below 1 unless they have been loaded with 0.

6.25. Y_amp.mag The Y_amp (figure 22) is a TCAMP resistor with an amplifier connected as a buffer. Input is a current from the DAC's; the output is a loadable voltage that should be able to drive over 20pF easily.

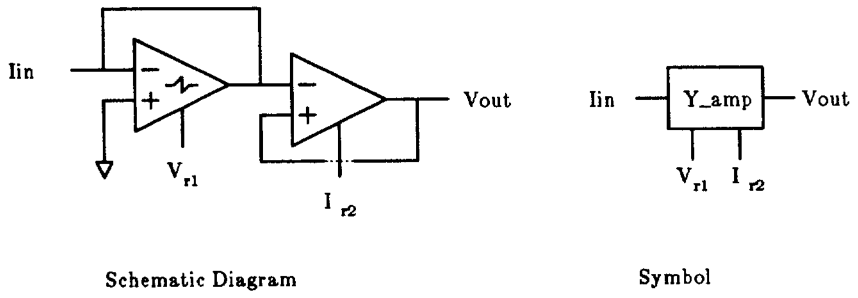


Figure 22 Y_amp

7. Summary

This chip will use mixed mode analog and digital circuitry to implement a Klopf-like circuit. The chip will have addressable inputs and weights, DAC's, op-amps, and nearly all components on board to make a newly stand-alone device. By using a decaying exponential, the learning rate constants can be removed.

Using capacitors as "memory" we have removed any need for register memory. A single KLOPF chip should show several conditioned reactions, while several linked together should show some higher order conditioning traits.

Through the use of various bias circuits, we should be able to completely test the analog circuitry. These supplementary voltages and currents will also enable us to "experiment" with our learning rule.

8. References

[1] Bhupendra K Ahuja, "An Improved frequency Compensation Technique for CMOS Operational Amplifiers", *IEEE J. Solid-State Circuits,* Vol. SC-18, No.6, December 1983.

[2] Klaas Bult and Hans Wallinga, "A CMOS Four Quadrant Muyltiplier", *IEEE J. Solid-State Circuits,* Vol. SC-21, No. 3, pp. 430-435, June 1986.

[3] Harry Klopf, "A Neuronal model of Classical Conditioning", AFWAL-TR-87-1139, October 1987.

[4] Carver Mead, *Analog VLSI and Neural Systems,* Preliminary 2 (10/8/1987), California Institute of Technology, 1987.