Discrete Transistor Logic for homemade CPUs

Jesus Arias-Alvarez

Dpto. E. y Electrónica, E.T.S.I. Telecomunicación. 47011 Valladolid. Spain

1 Introduction

The title of this document is an aberration. If somebody is building a CPU using discrete transistors today He is probably insane. Yet, there are some examples to find around the Net, like the Megaprocessor [1], where its designer went so nuts that he even built the memory using discrete transistors, or the Monster6502 [2], where a clone of the well known NMOS 6502 was built by following the reverse engineered schematic and by placing a discrete MOSFET in the position of any of the original accumulation transistors (depletion transistors were replaced by resistors). Its designer also kept the original floorplan, ending up with a board which is an scaled up version of the original chip die.

One must wonder about the motivations of these people to invest so many hours of work, not to mention the sheer cost of PCBs, thousands of components, and hundreds of meters of solder wire, and I suspect they did so because of:

- Nostalgia of the punched-car era, some sort of "steampunk computing".
- If it can be done somebody has to prove it.
- The quest to fully understand the most intricate details of processors that today remains hidden from almost everyone (take for instance the carry look-ahead logic).

Well, moving aside the motivations we can examine the way these CPUs were made and their performance in terms of cost and computing power. The Megaprocessor and Monster6502 are built around NMOS logic [3]. James Newman, the designer of Megaprocessor says he chose NMOS because they require one less resistor per input than a similar RTL gate (for RTL gates watch [4]). The Monster6502 was NMOS just because the original 6502 also was.

The problem with both processors is that they only allow for a 50kHz clock, witch is very slow even for the standards of the sixties. Eric Schlaepfer, the designer of the Monster6502 correctly blames the high gate capacitance of MOSFETs as the responsible of the low clock frequency. The fact is that these transistors, in spite of their small package, can switch much higher currents than those found in the gates. The gate capacitance is high because transistors are big.

There is an straight solution: Divide each pull-up resistor by 20. The gate current will be multiplied by the same factor and the Monster6502 will run at 1MHz. The only problem is that it will end burning more than a hundred watts. So the NMOS choice don't look very promising for a discrete transistor CPU. It is attractive because of its simplicity, but if you also want speed you must look for other alternatives.

It is worth mentioning the MT15 CPU [5], designed by Dieter Mueller. It uses bipolar transistors and the logic used was intended to be fast from the beginning. It is a sort of TTL that uses NPN and PNP transistors, diodes, resistors, and capacitors. Logic gates are complex but they are fast. The processor runs with a 500kHz

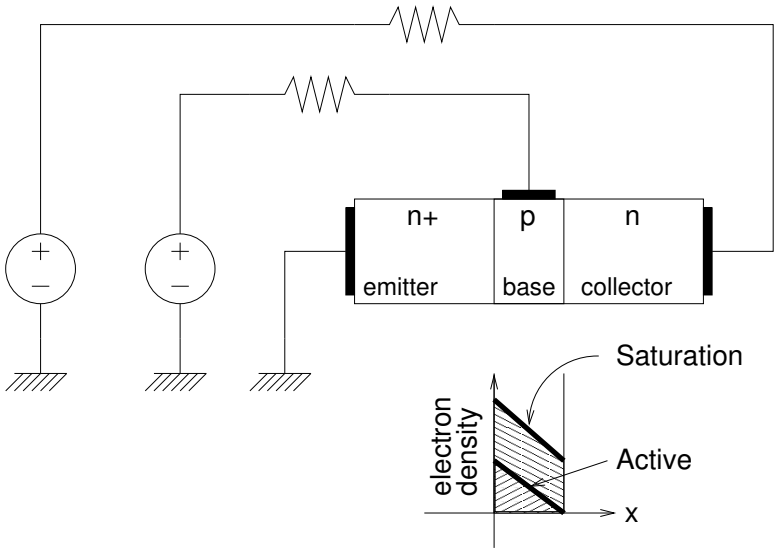

Figure 1: The charge stored in the base of a BJT depending of its operating region.

clock, but to be fair we have to take into account that one clock cycle is divided into 4 phases, so its master clock is 2MHz, 40 times faster than the NMOS counterparts. The logic runs with a 2.5Volt supply and the whole CPU consumes around 5Amps.

So the bipolar transistor seems to be the right choice for these kind of computers, and we have a pretty good design, the MT15, that can be used as a reference for benchmarking. My aim here is to study an alternative kind of logic hoping to achieve a faster speed and a lower component count. That logic is ECL [6], a now forgotten bipolar logic that decades ago powered the Cray supercompuer. ECL gates are complex, but when simplified to their bare minimum they even require less components than an equivalent MT15 gate. And they are faster while also requiring less current.

1.1 The BJT as a switch

The BJT transistor, when operated as a switch, usually changes its state from cut-off to saturation or in reverse. In the saturation state both the emitter-base and the collector-base junctions are forward biased, so there are charges being injected into the base region from both sides, and as a consequence a large amount of minority carriers (electrons for an NPN transistor with a P-type base) builds up in the base (see figure 1). The situation is different for the active state where the collector-base junction is reverse biased and any minority carrier that drifts close to the collector is sweep away by the electric field, resulting in a negligible minority carrier density in the collector side and in a total charge stored in the base that is much less than in the case of saturation.

When the BJT is turned off the charge stored in the base has to be drained away or the transistor will keep conducting for a long time. In that sense to left the base floating when the transistor is off is a bad idea. It is much better to tie the base to ground (or to Vcc for PNPs) through a low impedance resistor or even to a negative voltage in order to achieve a fast turn-off.

Also it would be better to avoid biasing the transistor in saturation in first place because in this case there is less charge to remove from the base. Yet it is difficult not to enter saturation when on with the usual switching circuits.

1.2 The BJT differential pair

ECL gates are based on differential pairs of BJTs. These circuits basically divide a constant current at the emitters into two different currents at the collectors. How much current goes to each collector depends on the voltage difference at the bases and little else (there is a dependence on temperature, but when temperatures are

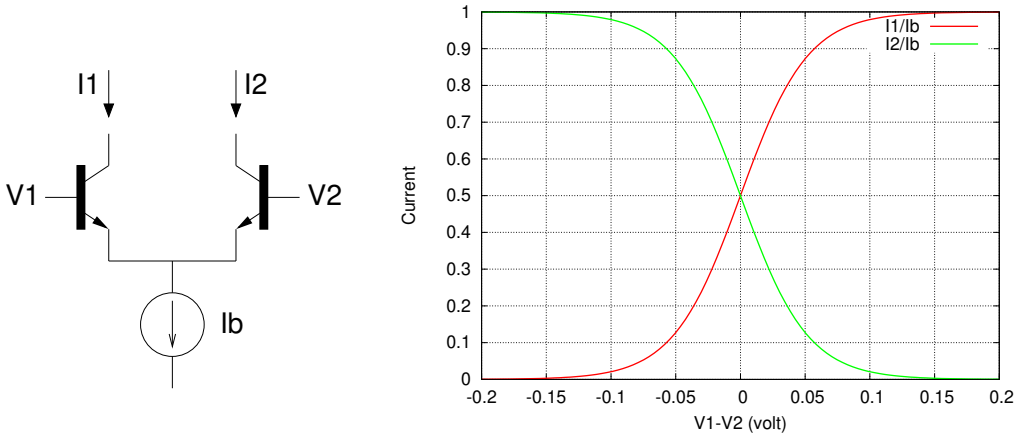

Figure 2: BJT differential pair and its transfer function

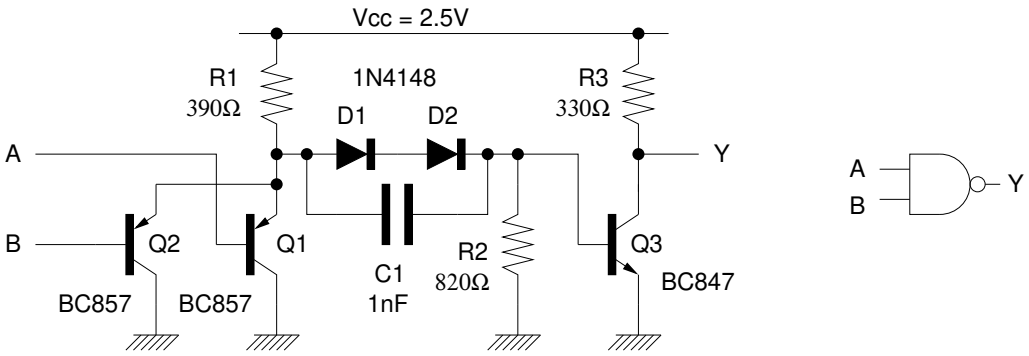

Figure 3: Dieter Mueller's NAND gate.

given in Kelvin the relative variation is small). In figure 2 the currents are shown as a function of the voltage difference. The curves are:

$$I_1/I_b = 1 - 1/(1 + \exp((V_1 - V_2)/V_T))$$

$$I_2/I_b = 1/(1 + \exp((V_1 - V_2)/V_T))$$

Where $V_T = KT/e$, with K being the Boltzmann constant, T, the absolute temperature in Kelvin, and e the charge of the electron. V_T is more or less 26mV at room temperature.

We can see that with only a 100mV difference at the bases almost all the current goes to just one collector (about 98% of the current). So for higher voltage differences the differential pair acts as a current switch, steering the emitter current towards one collector or the other.

2 Bipolar Logic gates

The basic gate of the Dieter Mueller TTL, or DMTTL, is the NAND gate shown in figure 3. Its logic levels are 2.5V when the output is high and less than 0.2V when the output is low. Its component count is:

#	Device	Comments
n	PNP (BC857)	1 transistor per input (n inputs)
1	NPN (BC847)	output switch
2	diodes (1N4148)	small-signal diodes for threshold shift
3	resistors	
1	capacitor	speed-up capacitor

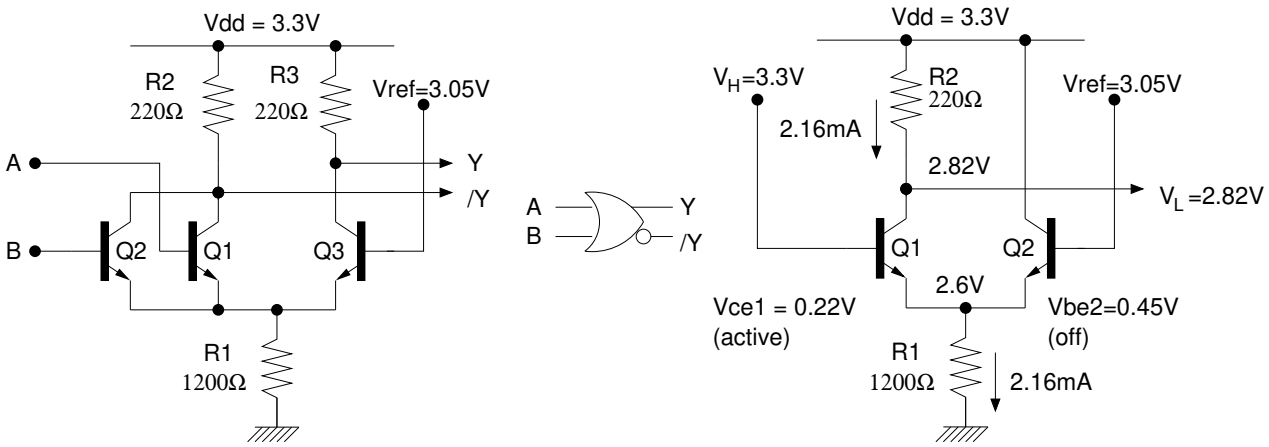

Figure 4: Proposed ECL OR-NOR gate and detail of transistor biasing (inverter gate).

The total count is 8 devices for an inverter plus one extra PNP transistor for each additional input in the case of a NAND gate.

In order to test this gate a ring oscillator of 3 inverters was simulated using Spice. Because there are one delay per inverter, and one signal cycle implies two changes, the average propagation delay of these gates will be 1/6 of the period of the oscillation. Also, the same circuit was built into a breadboard using the equivalent through-hole versions of the transistors (BC547 and BC557). The results were:

Average propagation delay (simulated)	4.6ns
Average propagation delay (measured)	11.1ns
Average current consumption	6.1mA (15.25mW)

The measured delay is much higher than the simulated one, but this comes at no surprise because we know the breadboard has a noticeable parasitic capacitance between contact rows (also, the Spice models for the transistors could have been too optimistic).

An “standard” 74LS04 will have a nominal 10ns delay and 2.4mA of current. Again, a ring oscillator was built using a 74LS04 IC and the measured average delay was just 5.1ns. In this case the measured delay is almost half the nominal value, meaning the datasheet is showing a really worst case value.

In summary, the speed achieved by the DMTTL gates is quite good, comparable to standard 74LS logic, yet they are a bit power hungry.

In these gates the output transistor goes into saturation when the output is low. I though about adding a Schottky diode between the base an collector of the NPN transistors in order to avoid their saturation [7]. The Schottky diode has a lower forward voltage than the collector-base junction of the transistor, so it will start conducting when the transistor is still in its active region taking away the excess base current that otherwise will put the transistor into saturation. This is done in the 74LS logic with good results, but in the DMTTL case the speed is almost the same as before. Probably the additional capacitance of the diode, a BAS40 model, with 120mA maximum continuous current, compensates any speed improvement due to the transistors remaining in the active region, so this modification is ineffective.

In summary, these are the numbers we are trying to improve:

- 8 devices per gate, plus (n-1) devices if there is more than one input.
- 4.6ns average delay.
- 15.25mW of power per gate.

The proposed schematic of the ECL gate, which I called Bare-Minimum ECL, is shown in figure 4. It differs from conventional ECL gates in several ways:

- There are no level shifting transistors at the output, so the high output voltage equals Vcc.
- The low level output is only 500mV bellow Vcc. This low voltage swing was chosen to ensure the output transistor is biased in its active region when the output is low.
- No negative voltage is used for the supply. But, we must remember that all signals are voltages referred to Vcc, not to GND. In that sense decoupling capacitors for Vref are better connected to Vcc instead of GND, and if a copper plane is used in the PCBs it is better to assign it to Vcc than to GND.
- A 3.3V was selected as the supply voltage instead of 5.2V for two reasons: first, today 3.3V are commonly used in many circuits instead of 5V, and second, the ECL gates are basically current steering circuits, meaning that what really matter are currents instead of voltages. And for a fixed current, the lower the voltage the lower the power dissipation.

Also notice that the BMECL gate provides us with two outputs, one with the NOR function and another with the OR function. These complementary outputs surely can save some inverters in our schematics, but if one of these functions is not needed its corresponding resistor can be replaced by a short to Vdd, saving one component in the bill of materials. In any case there are less components required than for a DMTTL gate:

Logic	#devices (2-input gate)	#transistors	#resistors	#other
DMTTL	9	3	3	3
BMECL	5	3	2	0

And now for the ring oscillator simulation and measurement. The results for the BMECL were:

Average propagation delay (simulated)	2.5ns
Average propagation delay (measured)	6.7ns
Average current consumption	2mA, (6.6mW)

Again, the measured delay more than doubles the simulated one and the same explanation can be given here. Comparing these results with those of the DMTTL logic we can see that we can almost achieve half the propagation delay with only 1/3 of the current and a 44% less components to solder on the board. Also in our case the transistors are all of the NPN type.

Now the bad news. While being simpler and faster than the DMTTL logic, the BMECL has its own inconveniences. Lets mention some of them:

- The voltage levels are incompatible with any other logic, particularly CMOS. Therefore level conversion circuits are required in order to interface CMOS devices, like 3.3Volt RAMs. CMOS to BMECL doesn't require adaptation (as long as the supplies are of the same voltage, 3.3V), but BMECL to CMOS is impossible without voltage amplification.
- A 3.05V reference voltage is needed. Fortunately the current drawn from the gates is small, only a base current in the 10 microamp range.
- Our basic gate is an OR/NOR gate. AND gates can't be built this way. This is not a big problem because we can resort to De Morgan's theorem [8] and replace all AND gates with NORs. Anyway, it is still tempting to stack transistors in search for the AND gate, and to found the guidelines for some other interesting circuits, like the XOR gate or the data latch (more about this next).

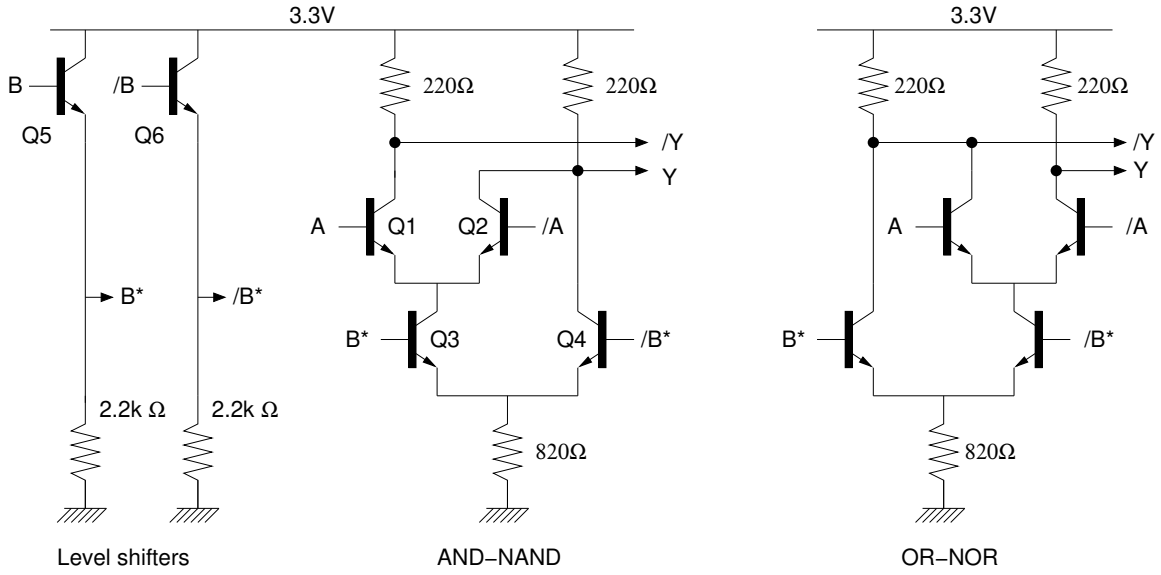

Figure 5: Differential gates

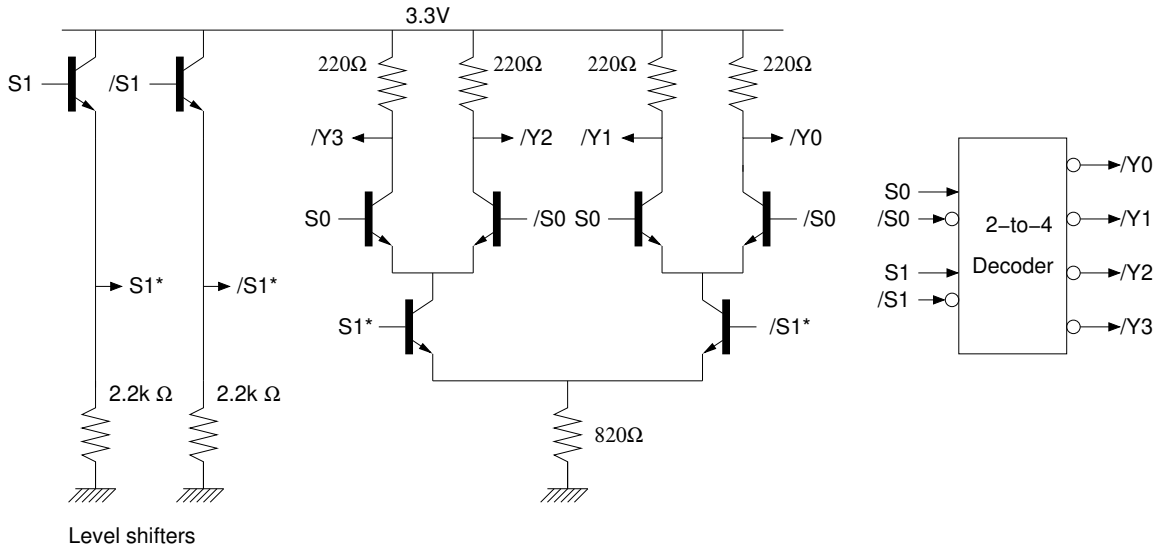

Figure 6: 2-layer implementation of a 2 to 4 decoder

3 Differential, Two-layer, BMECL

ECL gates provides us with a differential output and they can be modified to include also differential inputs. The trick is to stack differential pairs over differential pairs. In figure 5 a 2-input AND-NAND gate is built using this strategy. But we must be careful with the logic levels of the inputs of the lower transistor layer because the usual logic levels will saturate Q3 or Q4, so, the B and /B inputs have to be voltage shifted via the emitter followers Q5 and Q6.

The first circuit of figure 5 is an AND-NAND gate, but the same circuit can also be used to build an OR-NOR gate by just exchanging the inverted and non inverted inputs and outputs (that is: by applying the De Morgan's theorem)

The problem with these gates is the high device count. For a 2-input gate we need 4 transistors instead of 3, and assuming we already got the B an /B signals shifted, because otherwise another 2 transistors and resistors are needed. Also, it isn't trivial to expand the number of inputs: a 3-input gate will require another layer of transistors with more voltage shifting, so it starts to look impractical.

But, while of little interest for simple gates, the differential approach can be useful for more complex circuits. In figure 6 a 2 to 4 decoder is built using this kind of logic. The idea here is to have 4 possible

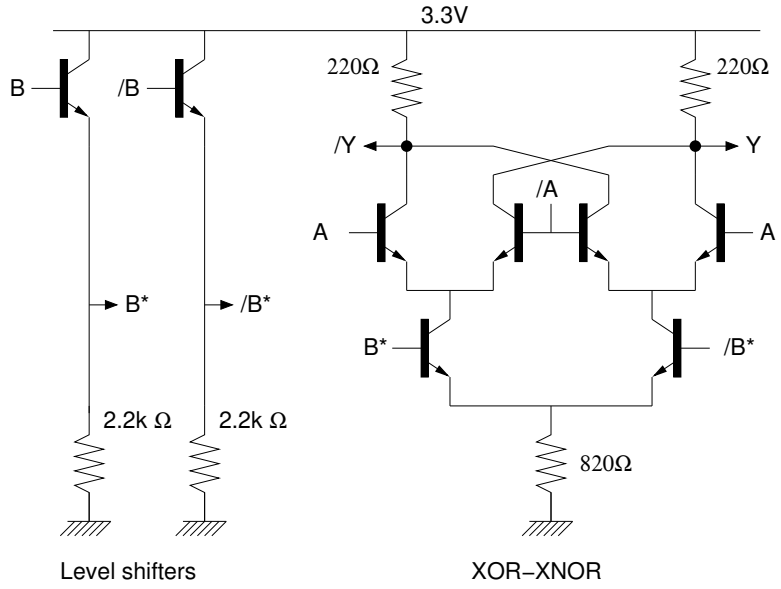

Figure 7: 2-layer implementation of a XOR-XNOR gate (Gilbert cell)

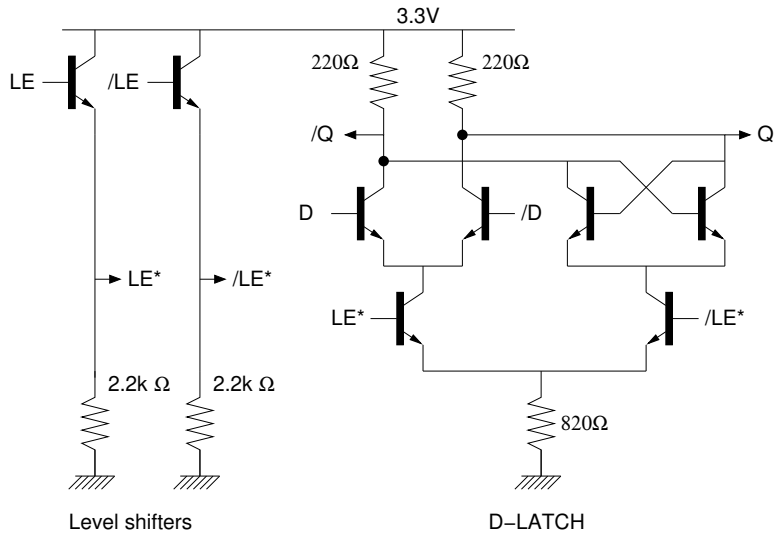

Figure 8: Data latch

paths for the current coming from the emitter's resistor, and only one output low (active). The device count is 8 transistors and 7 resistors (including level shifting), while a "by-the-book" design using four 2-input NOR gates would result in 12 transistors and 8 resistors. So, for this case the 2-layer differential circuit really saves components from the bill.

A 3-layer circuit for a 3 to 8 decoder will require 18 transistors, 13 resistors, and 2 diodes (for additional voltage shift), while a conventional circuit with 8 3-input NOR gates will end up using 32 transistors and 16 resistors. Here the saving is bigger than before, but a 3-layer circuit is really our stacking limit considering the voltage of the supply.

Another interesting circuit is the XOR-XNOR gate shown in figure 7. This circuit is probably very familiar for RF circuit designers (Gilbert cell [9]), where it was used as a mixer. Here the same circuit performs a logic function. The device count is 8 transistors and 5 resistors. An equivalent implementation using three 2-input NOR gates would result in 9 transistors and 6 resistors, so the Gilbert cell offers an small saving in terms of components, but it is also faster than a cascade of gates.

Another very interesting 2-layer circuit is the data latch shown in figure 8. In this case isn't very fair to count the components of the level shifters as part of the latch because a single shifter usually gets connected to a large number of latches. This reduces the device count to 6 transistors and 3 resistors. An equivalent circuit

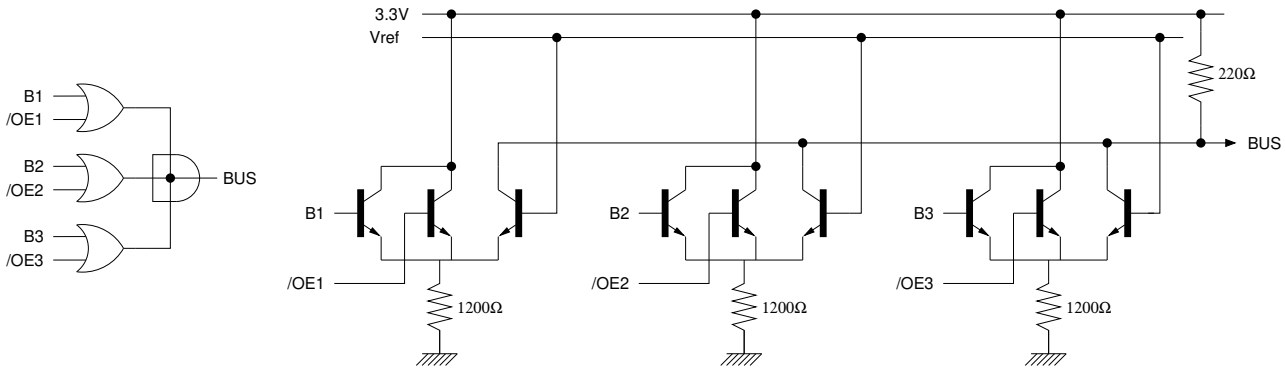

Figure 9: Output multiplexing using wired-AND logic and its detailed circuit

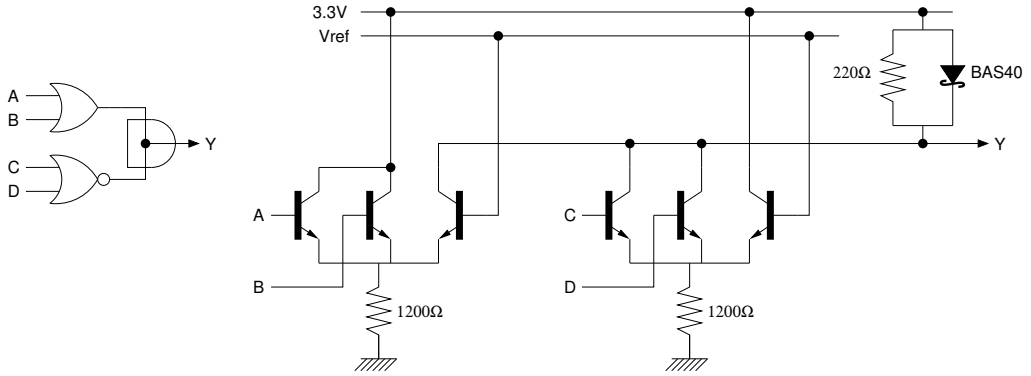

Figure 10: Wired-AND example with voltage clamping

using four 2-input NOR gates would require 12 transistors and 8 resistors, so, the 2-layer approach halves the device count.

A cascade of two latches, with opposite enable signals, will form a master-slave flip-flop, resulting in a device count of 12 transistors and 6 resistors, which is comparable to the 16 transistors of a CMOS flip-flop. Such a flip-flop, built with BC547 transistors, was simulated using SPICE and it was capable of toggling with a 20MHz clock frequency.

Also it is worth mentioning that for these differential circuits a differential signal isn't always needed at the inputs. The complementary input can be replaced with Vref and circuits will still run.

4 Wired-AND and multiplexing.

DMTLL logic uses the wired-AND trick very often in order to reduce the gate count as much as possible. BMECL can also resort to the same trick to some extent. The differential pairs have a current output and current outputs can be shorted together without resulting in excessive current consumption. But we must also be aware of the total current sum or we can end up with an invalid logic level. As an example, in figure 9 a data multiplexer is built by connecting the outputs of three OR gates to a single pull-up resistor. This circuit only works as intended if only one of the three /OEx signals is low at any given time. This condition guarantees a current in the pull-up resistor of 0mA or 2.16mA, but no more. Notice that if two or three /OEx signals are low the current can raise up to 6.48mA and the logic level at the output would be too low, driving the output transistors into saturation.

In order to have a more general wired-AND logic we can resort to the addition of Schottky diodes in parallel with the pull-up resistors (see figure 10). The forward voltage of these diodes is around 500mV and they will clamp the low logic level when excess current is directed to the output, thus avoiding the saturation of the output transistors.

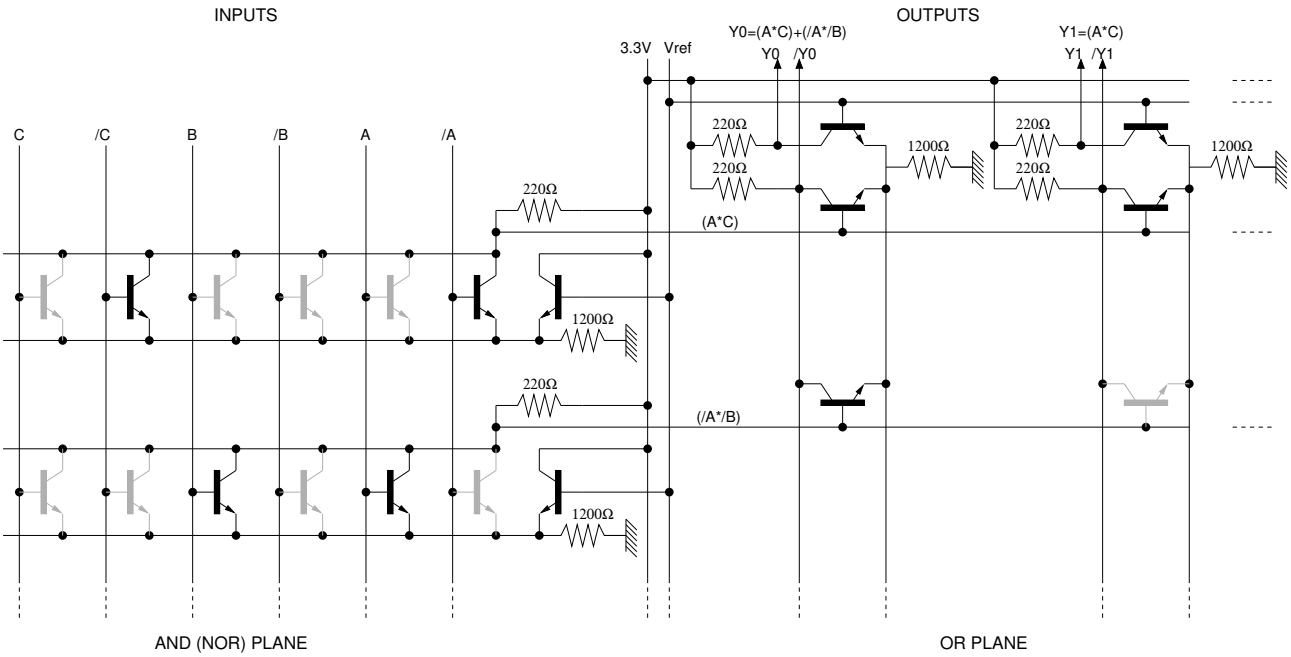

Figure 11: Schematic of a BMECL PLA. Shaded transistors aren't present.

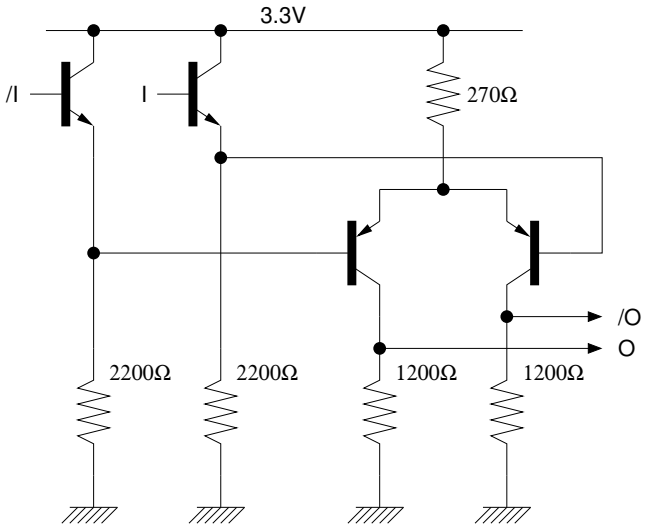

Figure 12: Interface from BMECL to 3.3V CMOS

5 PLAs

Programmable Logic Arrays, or PLAs, are logic functions of input variables arranged as two arrays of transistors. The first array, properly called AND-plane, provides us with the logic product (AND) of any desired combination of inputs. In our BMECL implementation the AND function is replaced by a NOR, according to De Morgan's theorem. The NOR gates of the AND plane have a variable number of inputs determined by the presence or absence of the corresponding transistors in the array (see figure 11).

After the AND plane follows an OR plane built in the same way. PLAs can be built as empty templates in a PCB and then programmed by the soldering of the required transistors.

6 Interfacing

The BMECL signals are of low amplitude and this makes them difficult to adapt to other circuits, particularly CMOS. The input is not problem and a 3.3V CMOS signal can be connected directly to a BMECL input, but

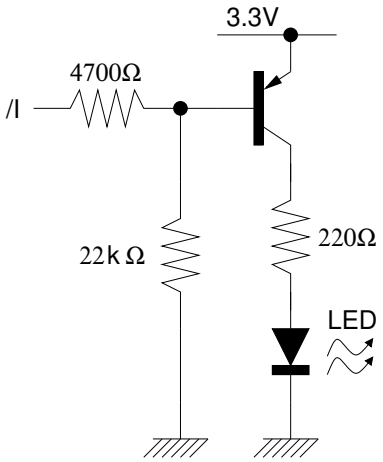

Figure 13: Interface for LEDs

BMECL outputs have to be amplified before applying them to CMOS inputs. In figure 12 a level converter circuit for this purpose is shown. First, the inputs signals are shifted, and then amplified. At the output a high level is now around 2.5V, that is enough to be read as a logic high by a 3.3V CMOS device. This circuit is also quite fast as no transistor ever enters saturation. Also notice that PNP transistors are being used for first time.

A CPU built with discrete transistors will probably also include a lot of LEDs to show the status of registers visually. The BMCEL signals don't have enough amplitude to lit a LED, but in this case speed isn't required and we can resort to the simple interface of figure 13. Here the PNP transistor goes into saturation and drives around 5mA of current through the LED when the input is low (2.8V at the input). Blue or white LEDs can't be used in this circuit because their forward voltage is near 3.2V. If used they will still turn on, but with less current than expected they could stay too dim.

7 Conclusions

In this paper a fast logic using small-signal discrete transistors is presented. The performance of some of these circuits was simulated using Spice and results look very promising, achieving timings comparable to those of the integrated 74LS logic and possibly faster. When compared with the logic of Dieter Mueller's MT15 [5], the proposed logic can result in a faster implementation while also requiring less components and less power to run.

References

[1] http://www.megaprocessor.com/

[2] https://monster6502.com/

[3] https://en.wikipedia.org/wiki/NMOS_logic

[4] https://youtu.be/sTu3LwpF6XI

[5] http://www.6502.org/users/dieter/mt15/mt15.htm

[6] https://en.wikipedia.org/wiki/Emitter-coupled_logic

[7] https://en.wikipedia.org/wiki/Baker_clamp

[8] https://en.wikipedia.org/wiki/De_Morgan%27s_laws

[9] https://en.wikipedia.org/wiki/Gilbert_cell