

10bit Cyclic ADC 设计

樊子辰 无 54 2015011065

谭淞耀 微 51 2015011075

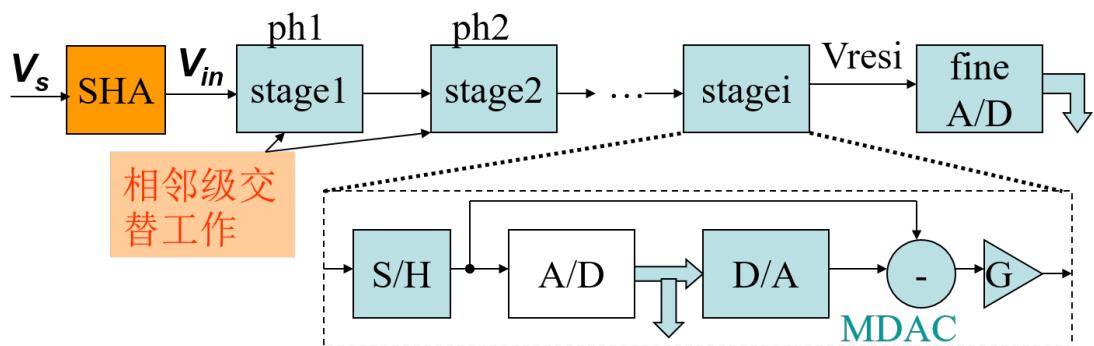
陆锦鹏 微 51 2015011200

张琳佩 微 51 2015011172

2019 年 1 月 20 日

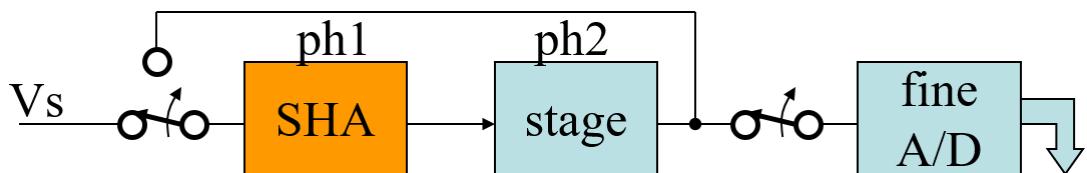
一 设计原理

如下图为 ADC 的基本原理框图。利用多步转换的原理，即将 ADC 分成几个子 ADC 进行比较，放大，做差等操作。并结合流水线设计，在多步转换结构中插入采样和保持中间结果的电路，即对模拟信号的采样保持电路(S/H)，来实现高速的流水线转换结构。使得相邻级交替工作，从而提升电路效率。



流水线 ADC

级电路由 S/H、A/D、D/A、模拟减法和模拟放大等功能电路构成，在开关电容电路实现中，其中的 S/H、D/A、模拟减法和模拟放大可由一 MDAC 电路来实现。流水线结构 ADC 由 SHA 和多个级电路级联而成，由于其具有对单个输入电压串行转换、对多个输入电压并行操作的特点，因此兼具多步转换的电路规模小和单步转换的速度快的优点。流水线转换结构是一种特别重要的 ADC 结构，广泛应用于高速高精度的 ADC 设计中。



循环式 ADC

在流水线结构中，各个级电路的实现结构都是一样的，利用这一点，我们可将多步的转换处理在同一个级电路上循环进行，由此可在流水线转换结构的基础上引入所谓的循环式转换结构，如上图所示。循环式 ADC 实现了速度与面积之间的交换，在某些对采样率要求不是很高但对面积要求很严格的应用场合，循环式结构是一种很好的选择。

二 电路设计

1. 放大器设计

根据 ADC 需要达到的分辨率、精度和采样频率的要求，以及跨导放大器的直流增益和增益带宽积与误差之间的关系，可以得到放大器的部分设计要求。

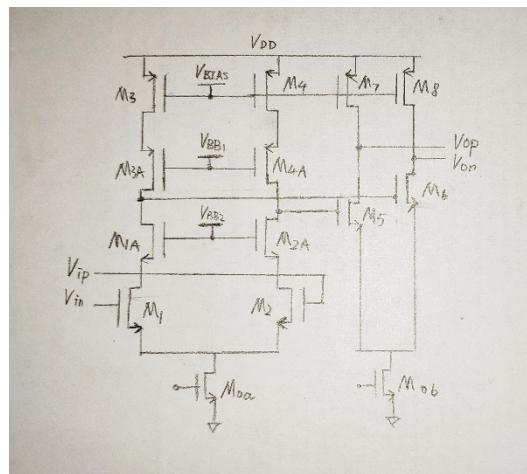
计算可得，MDAC 所用的跨导放大器需要达到的要求为直流增益 $A_0 \geq 80\text{dB}$ ，增益带宽积 $\text{GBW} \geq 100\text{MHz}$ 。

常见的跨导放大器结构有 telescopic 放大器、folded-cascode 放大器、两级放大器等。然而以上结构的放大器只能提供 $(gmro)^2$ 量级的直流增益，如果要达到 $A_0 \geq 80\text{dB}$ (10^4) 的条件，那么 $gmro$ 需要达到 100 以上。根据以往的经验，晶体管的尺寸可能要取极大的值。因此，必须使用具有更高直流增益的放大器结构（至少为 $(gmro)^3$ 的量级）。

根据已有的设计经验，我们选择使用改进的两级放大器结构，其中第一级为 telescopic 放大器，第二级为共源放大器。由下式：

$$A_{v0} = G_{mI}(g_{m1A}r_{o1A}r_{o1} || g_{m3A}r_{o3A}r_{o3}) \cdot G_{mII}(r_{o5} || r_{o7})$$

可得 $gmro$ 只需达到 28 以上即可满足上述设计要求。该设计符合简单两级放大器的分析方法与设计流程，只需要让每级放大器达到其设计要求，那么总体设计要求就可以达到。根据 0.18um 工艺下不同够到长度的 $gmro \sim vds$ 曲线图，可以确定 n 管和 p 管的沟道长度。本设计中 C_s 与 C_f 均为 0.5fF。根据电容值和频率特性等指标，用 gm/Id 的设计方法，确定流过晶体管的电流和各个晶体管的尺寸。偏置电路方面，使用简单的 mos 电流镜与低压 cascode 电流镜来提供合适的偏置。由于共模特性对电路性能有较大影响，为使电路稳定输出，需要增加共模反馈电路。由于本设计存在现成的两相不交叠时钟，可以使用开关电容型共模反馈环路。



最初设计的放大器主级电路

实际调试时，一开始使用 HSPICE 进行仿真，发现晶体管的直流工作点无法达到预期。调节参数后，电路性能未能达到预期；后来发现 HSPICE 所使用的工艺库为以前使用过的 0.35um 工艺库，最后没有办法达到想要的效果。后用 cadence 搭建原理图进行仿真，发现单独对第一级进行考察时基本满足要求，但加上第二级后整体的直流增益和频率特性变得十分差。经过多次参数调整，最终仍无法得到满足设计要求的电路。

最终仿真时采用的放大器为 verilog-A 模型，因为 ahdLib 库中只有单级输出的放大器，故设计差分输出的放大器，代码如下：

```

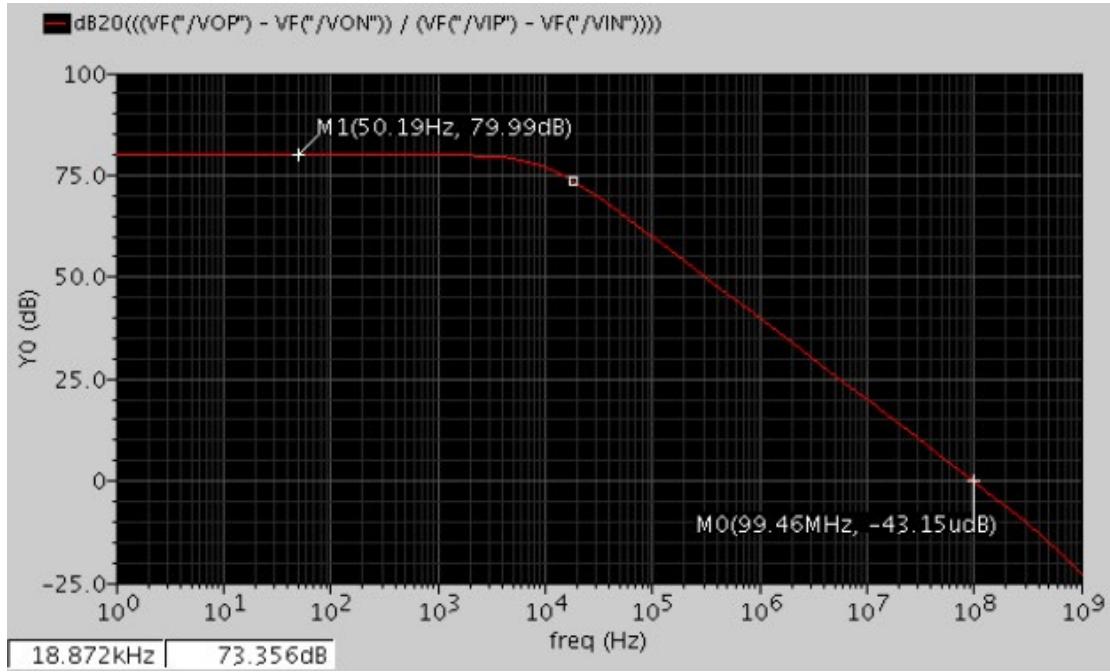
module opamp_diff( vout_p, vout_n, vref, vin_p, vin_n, vsupply_p, vsupply_n );
  input vref, vsupply_p, vsupply_n;
  inout vout_p, vout_n, vin_p, vin_n;
  electrical vout_n, vout_p, vref, vin_p, vin_n, vsupply_p, vsupply_n;
  parameter real gain = 835e3;
  parameter real freq_unitygain = 1.0e6;
  parameter real rin = 1e6;
  parameter real vin_offset = 0.0;
  parameter real ibias = 0.0;
  parameter real iin_max = 100e-6;
  parameter real slew_rate = 0.5e6;
  parameter real rout = 80;
  parameter real vsoft = 0.5;
  real c1;
  real gm_nom;
  real r1;
  real vmax_in;
  real vin_val;
  electrical cout_p, cout_n;

analog begin
  @ ( initial_step or initial_step("dc") ) begin
    c1 = iin_max/(slew_rate);
    gm_nom = 2 * `PI * freq_unitygain * c1;
    r1 = gain/gm_nom;
    vmax_in = iin_max/gm_nom;
  end

  vin_val = V(vin_p,vin_n)/2 + vin_offset;
  //
  // Input stage.
  //
  I(vin_p, vin_n) <+ (V(vin_p, vin_n) + vin_offset)/ rin;
  I(vref, vin_p) <+ ibias;
  I(vref, vin_n) <+ ibias;
  //
  // GM stage with slewing
  //
  I(vref, cout_p) <+ V(vref, cout_p)/100e6;
  I(vref, cout_n) <+ V(vref, cout_n)/100e6;
  if (vin_val > vmax_in) begin
    I(vref, cout_p) <+ iin_max;
    I(vref, cout_n) <+ -iin_max;
  end
  else if (vin_val < -vmax_in) begin
    I(vref, cout_p) <+ -iin_max;
    I(vref, cout_n) <+ iin_max;
  end
  else begin
    I(vref, cout_p) <+ 1*gm_nom*vin_val ;
    I(vref, cout_n) <+ -1*gm_nom*vin_val ;
  end
  //
  // Dominant Pole.
  //
  I(cout_p, vref) <+ ddt(c1*V(cout_p, vref));
  I(cout_p, vref) <+ V(cout_p, vref)/r1;
  I(cout_n, vref) <+ ddt(c1*V(cout_n, vref));
  I(cout_n, vref) <+ V(cout_n, vref)/r1;
  //
  // Output Stage.
  //
  I(vref, vout_p) <+ V(cout_p, vref)/rout;
  I(vout_p, vref) <+ V(vout_p, vref)/rout;
  I(vref, vout_n) <+ V(cout_n, vref)/rout;
  I(vout_n, vref) <+ V(vout_n, vref)/rout;
  //
  // Soft Output Limiting.
  //
  if (V(vout_p) > (V(vsupply_p) - vsoft))
    I(cout_p, vref) <+ gm_nom*(V(vout_p, vsupply_p)+vsoft);
  else if (V(vout_p) < (V(vsupply_n) + vsoft))
    I(cout_p, vref) <+ gm_nom*(V(vout_p, vsupply_n)-vsoft);
  if (V(vout_n) > (V(vsupply_p) - vsoft))
    I(cout_n, vref) <+ gm_nom*(V(vout_n, vsupply_p)+vsoft);
  else if (V(vout_n) < (V(vsupply_n) + vsoft))
    I(cout_n, vref) <+ gm_nom*(V(vout_n, vsupply_n)-vsoft);
end
endmodule

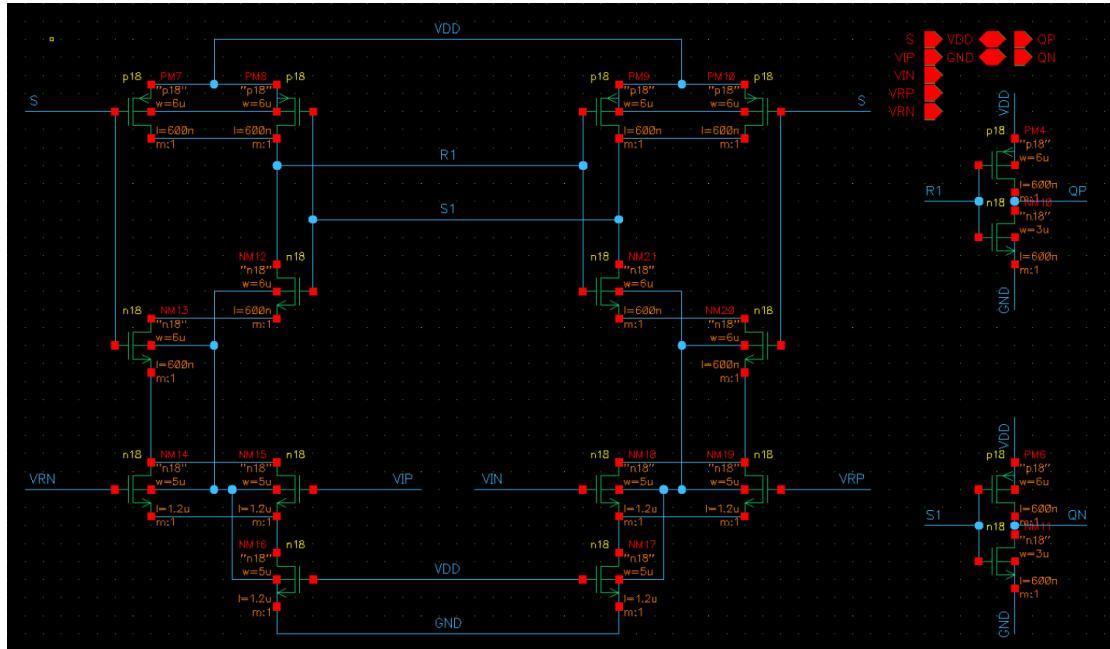
```

可以设计增益为 80dB, GBW 为 100MHz 以满足 ADC 需求, 仿真结果如下:



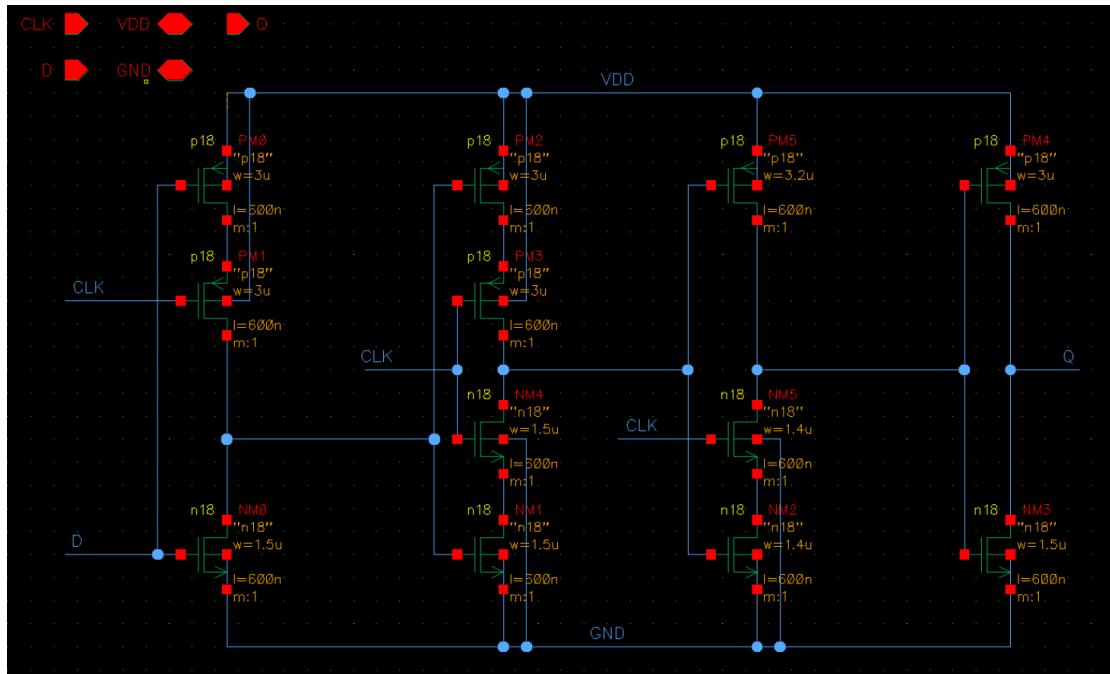
2. 比较器设计

比较器设计原理图如下：采用 Dynamic Comparator 结构，只是在触发比较瞬间消耗电流，在保持期间不消耗电流，故具有低功耗的优点。

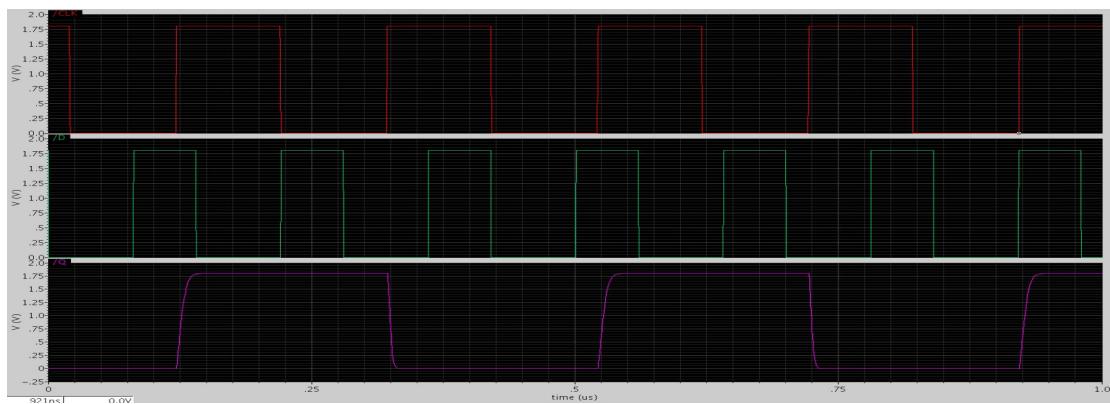


3. D 触发器设计

DFF 设计原理图如下：采用真正单相时钟 DFF，优点是面积小，速度快，且只要一个触发时钟；是一种主从 DFF，其信息在写到输出端以前暂存在寄生电容上，因此要求时钟具有比较陡峭的上升和下降沿，典型的最大沿上升和下降时间为 20ns 左右。

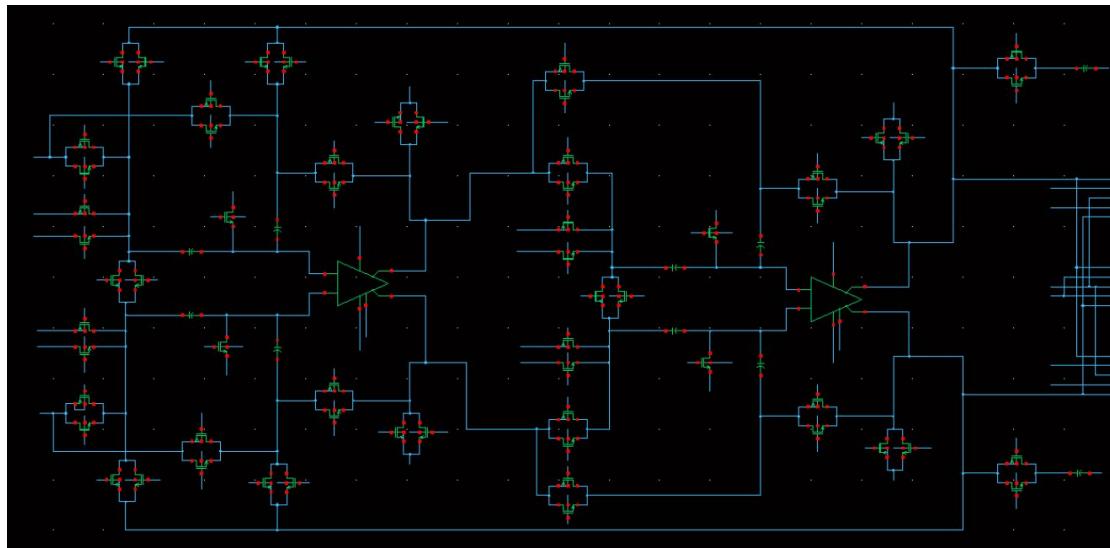


仿真结果如下：可知符合要求



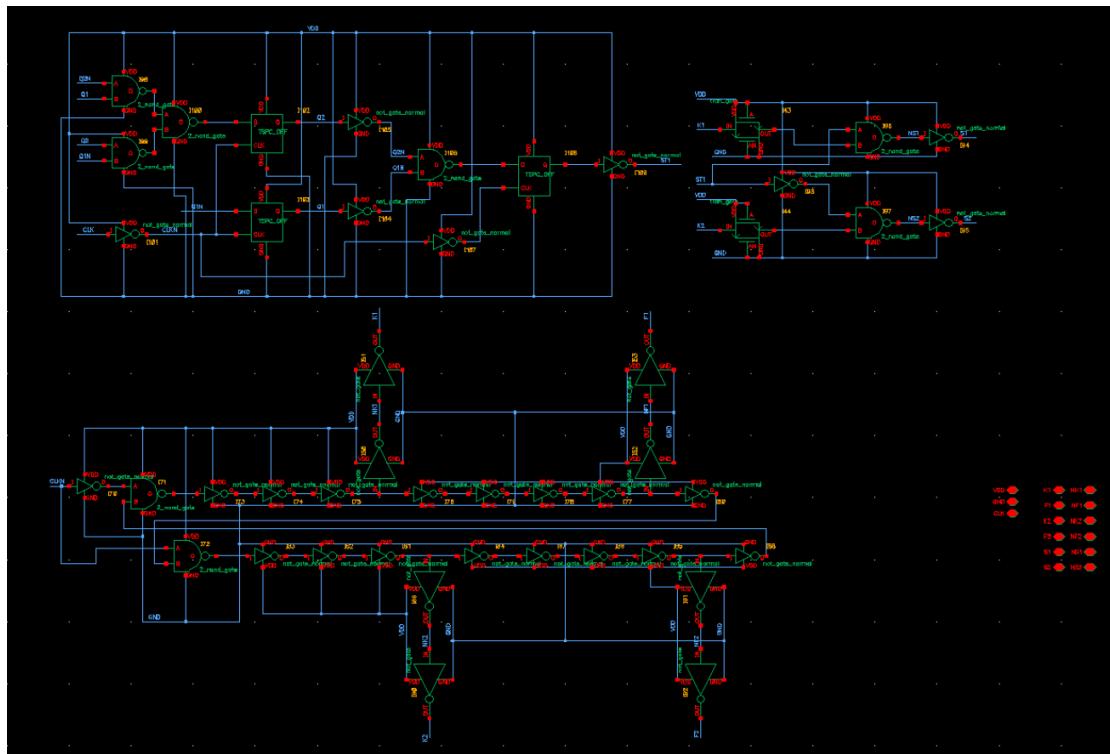
4. MADC 设计

MADC 原理图如下：其中的开关都采用最简单的晶体管传输门形式，其中开关的控制信号控制着 ADC 转换的周期。



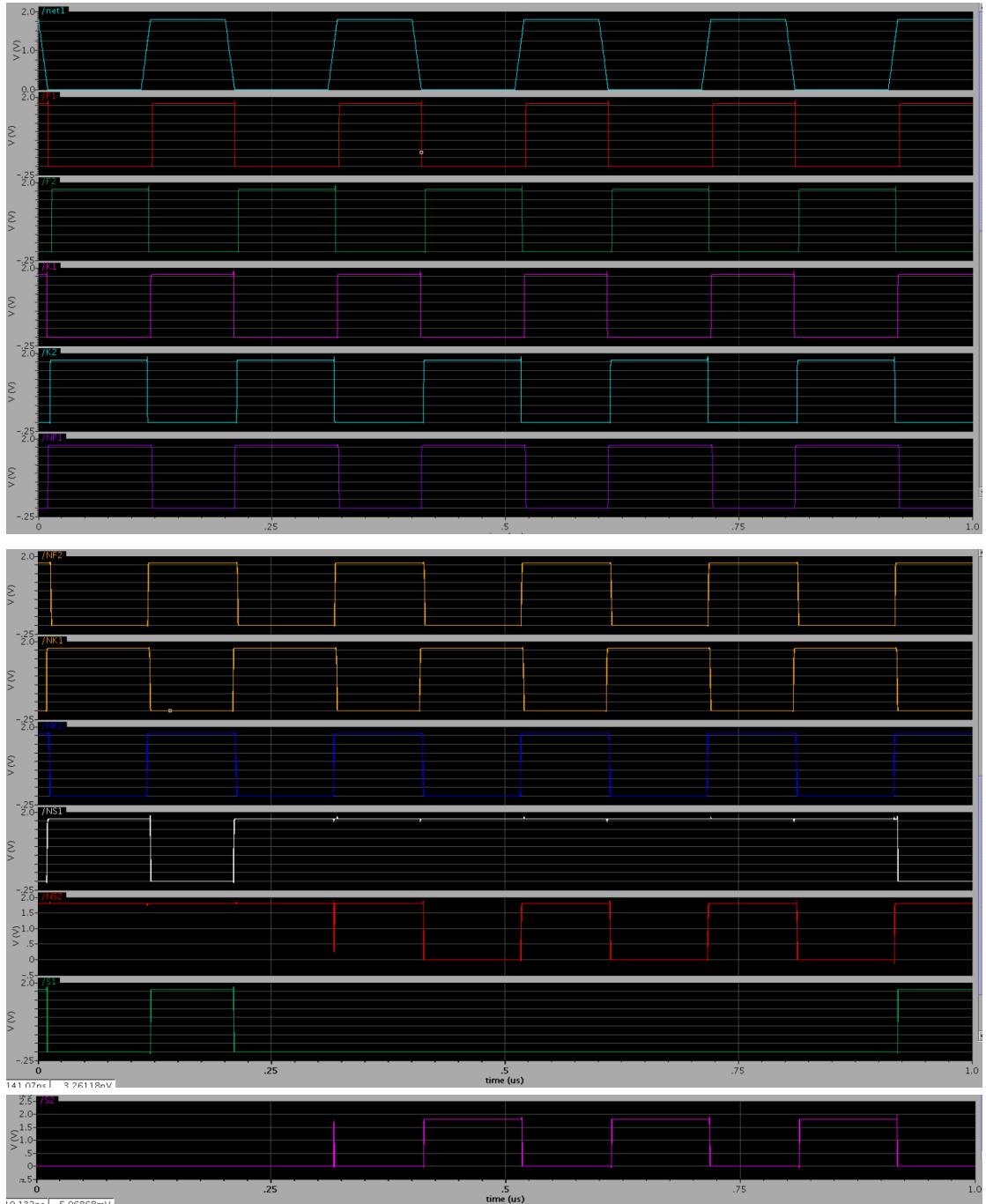
5. 时钟产生电路设计

时钟产生电路原理图如下：



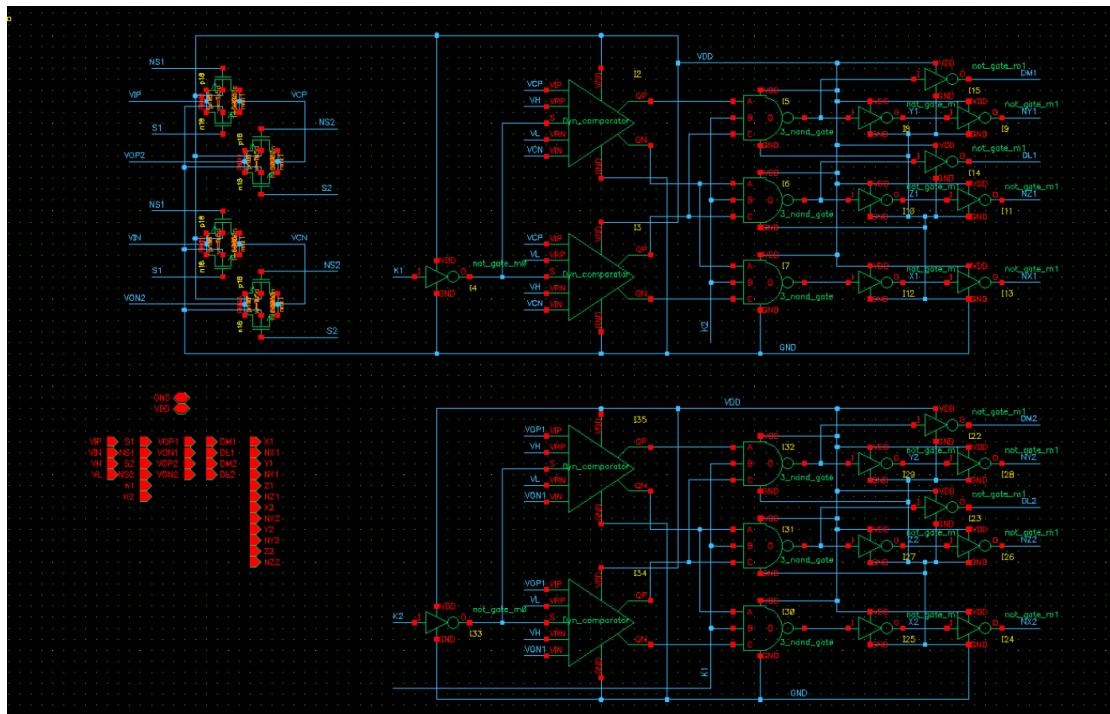
时钟产生电路仿真结果如下：

波形均符合预期。



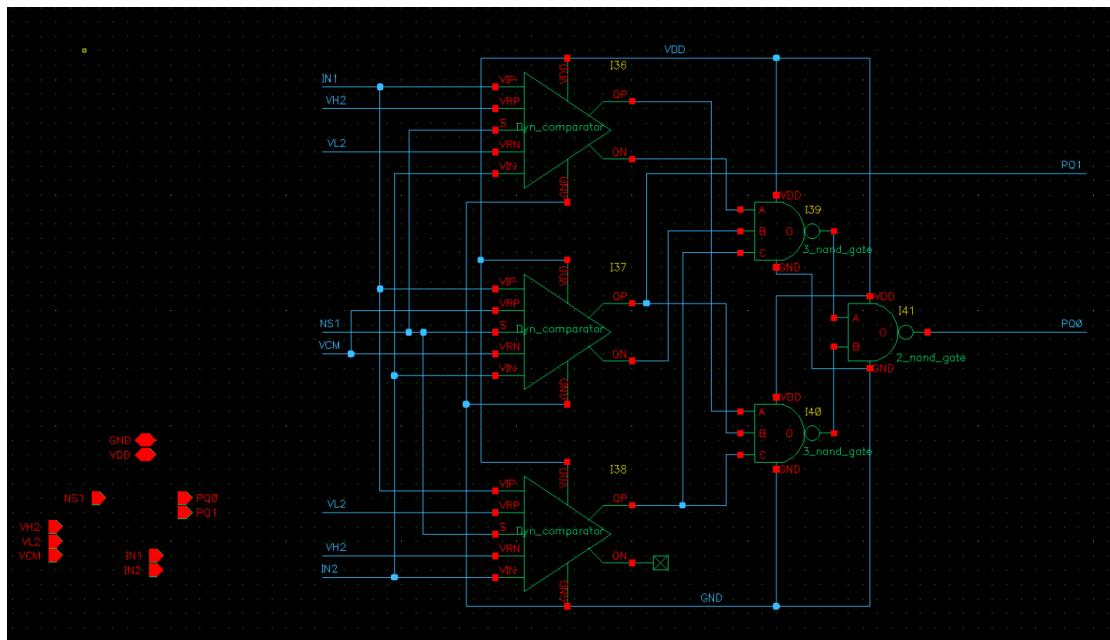
6. SubADC 设计

两个 1.5 bit subADC 电路原理图如下：



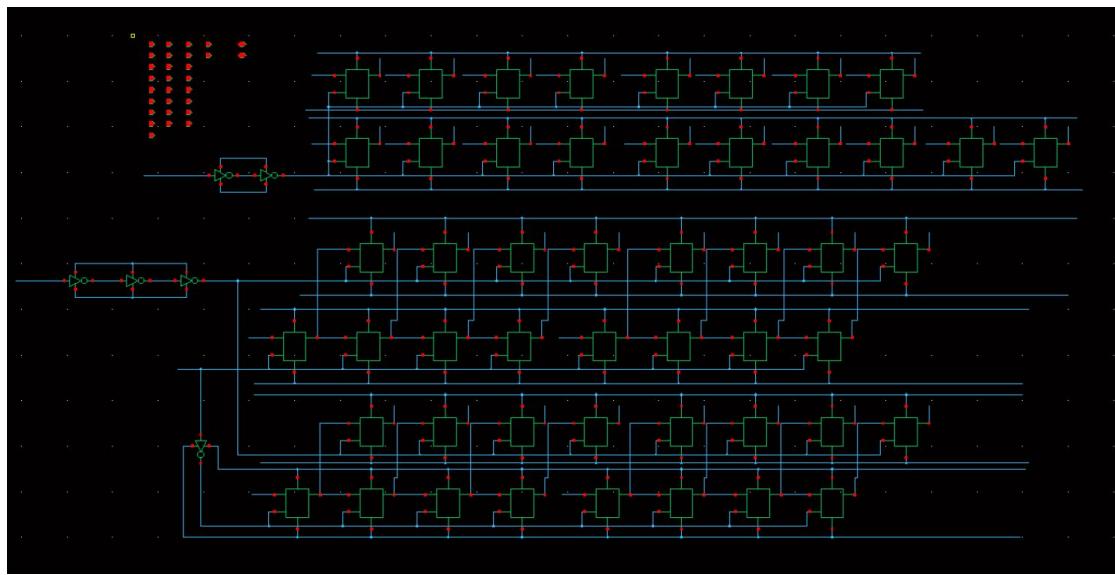
7. BackendADC 设计

BackendADC 设计原理图如下：



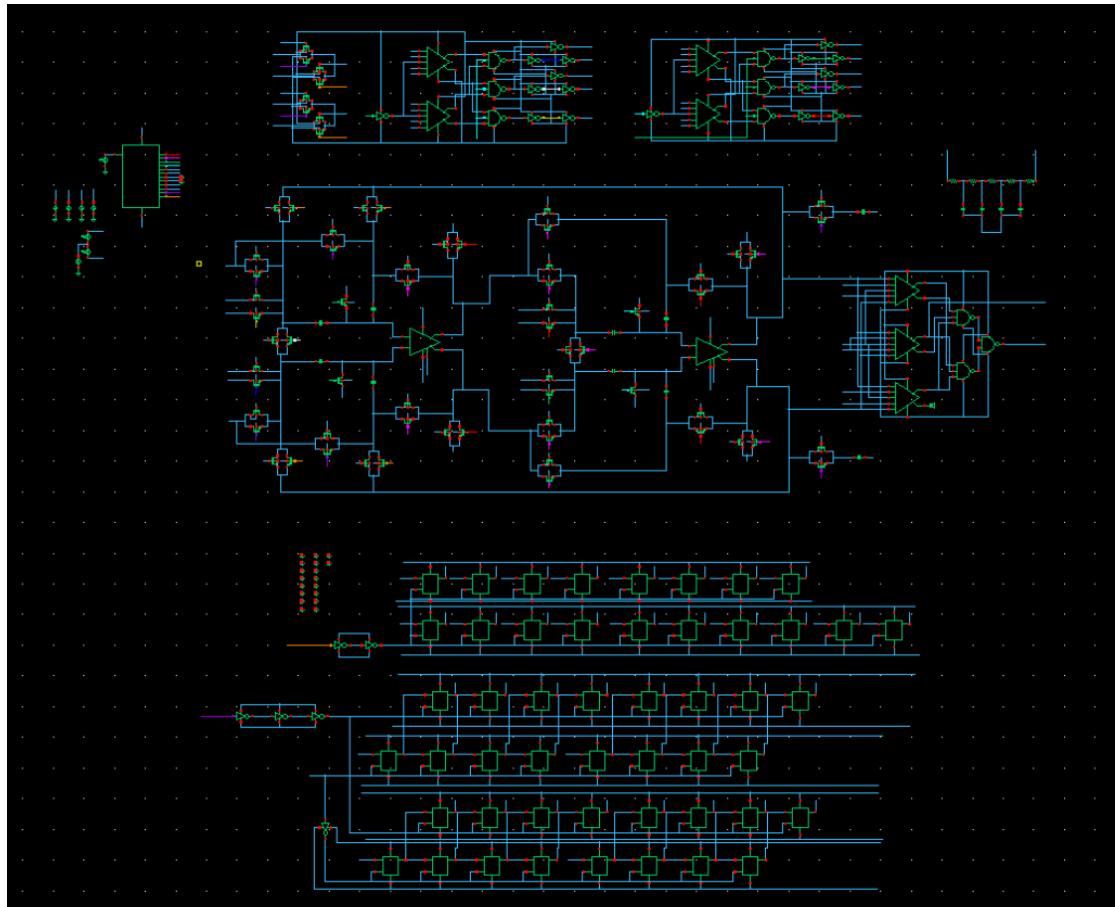
8. 同步输出电路设计

同步输出电路设计原理图如下：

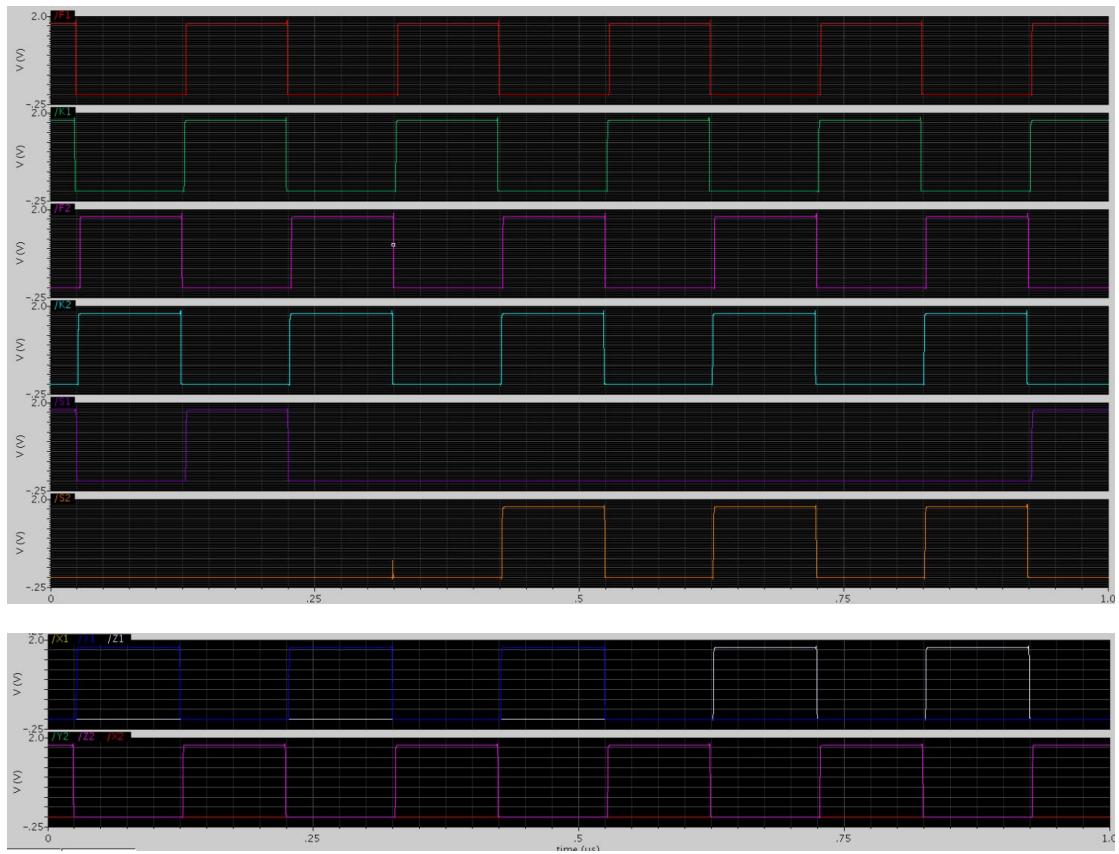


9. 整体设计

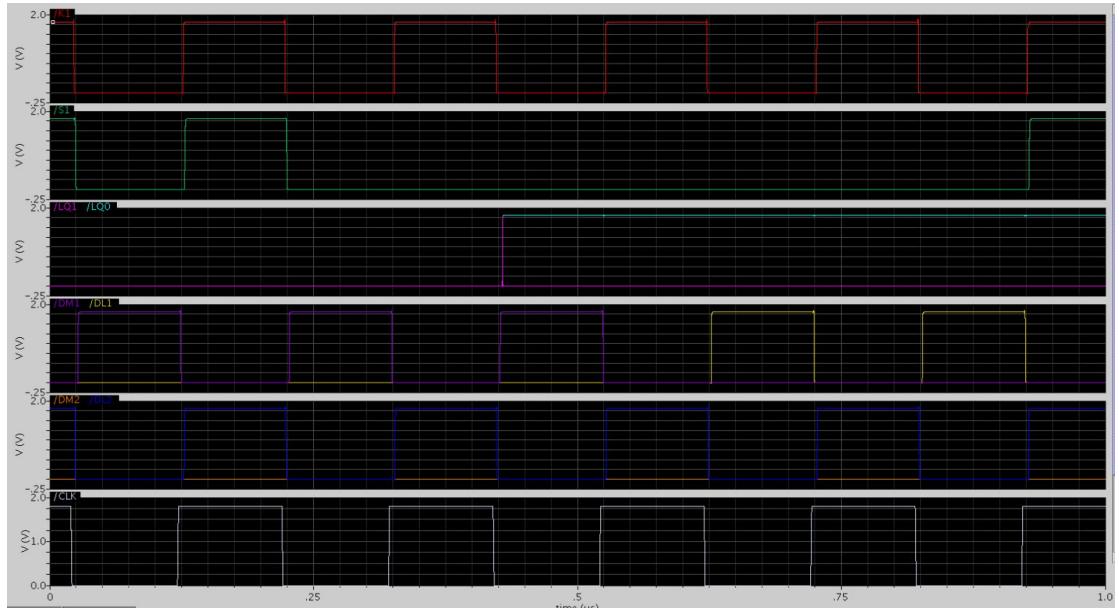
整体设计原理图如下：



级电路部分控制时序仿真波形如下：



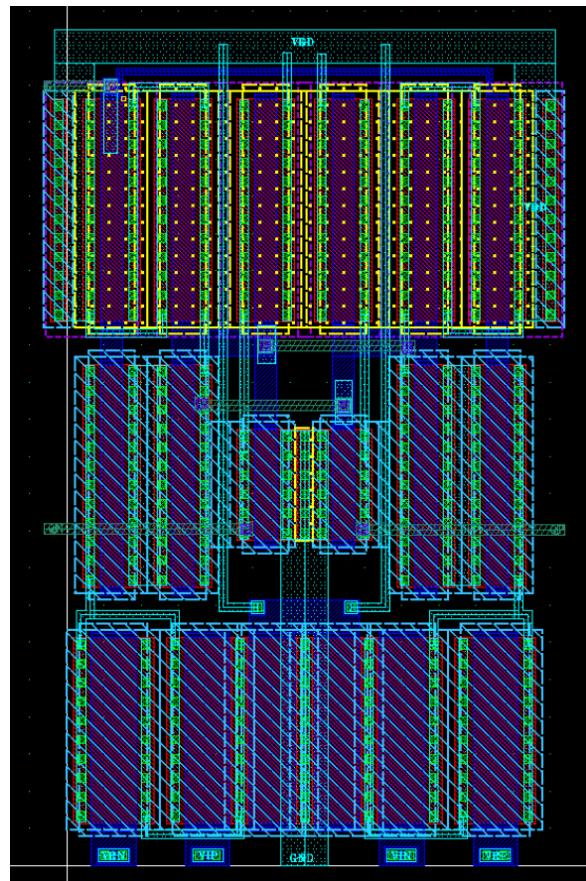
转换输出时序仿真波形如下：



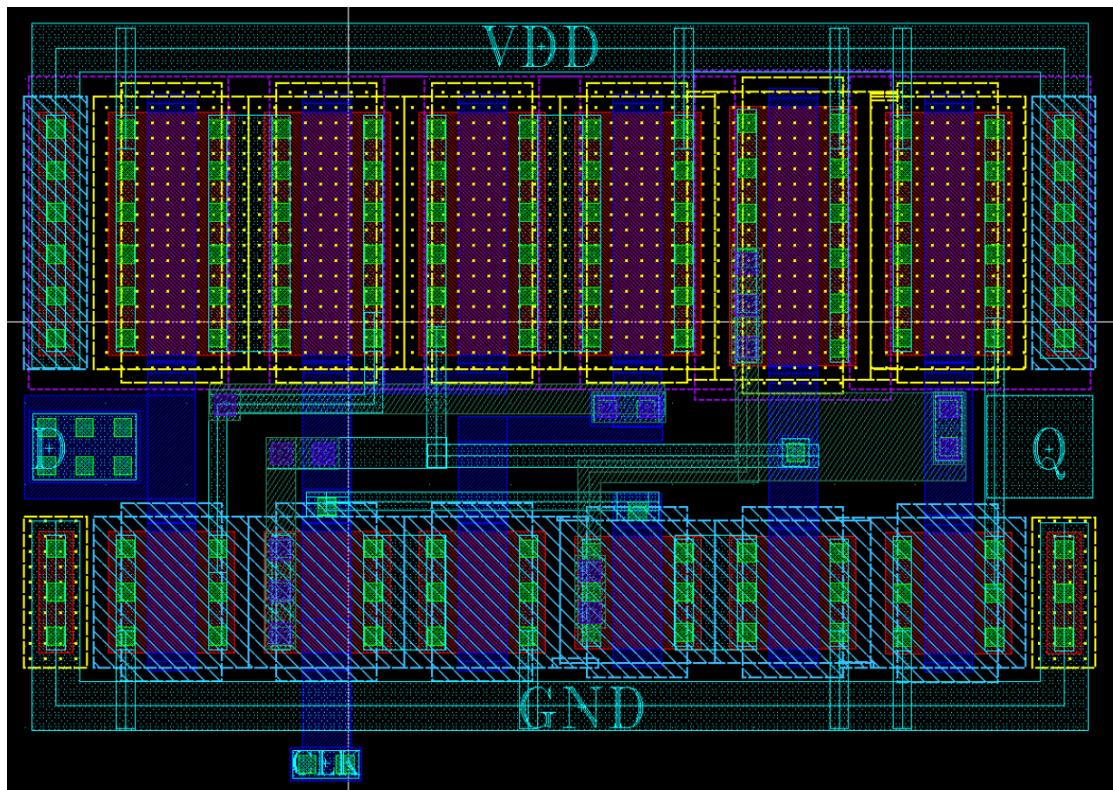
三 版图设计

(注：以下版图设计均通过了 LVS 和 DRC)

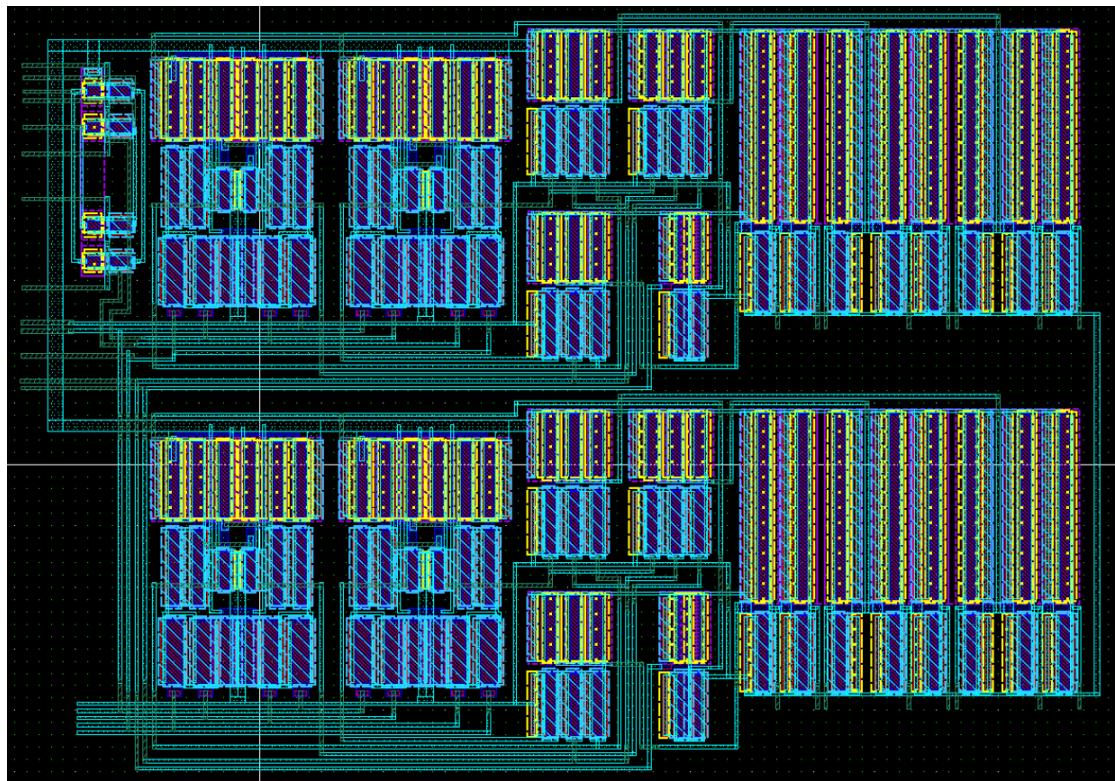
1. 比较器版图



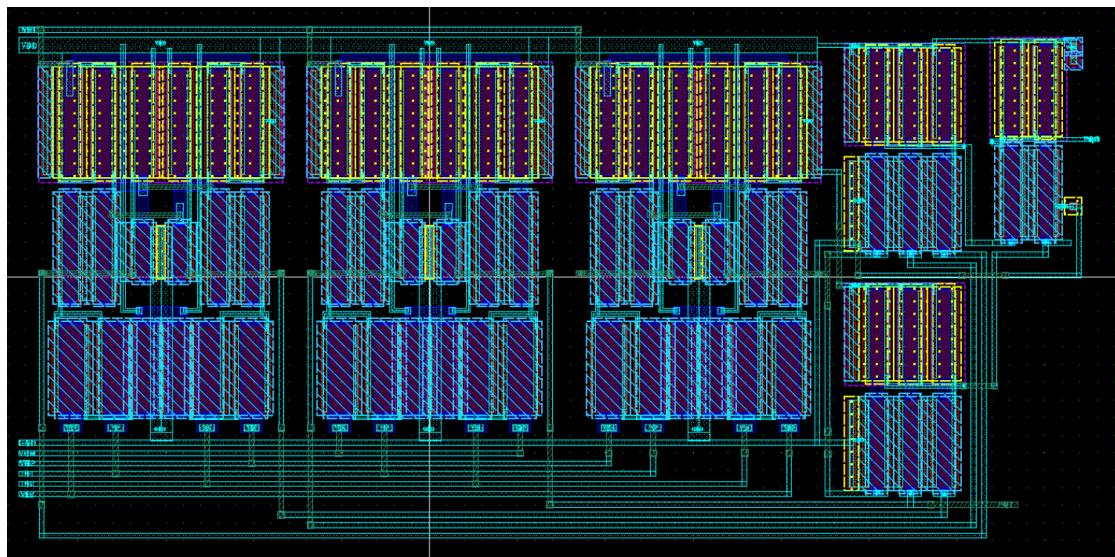
2. DFF 版图



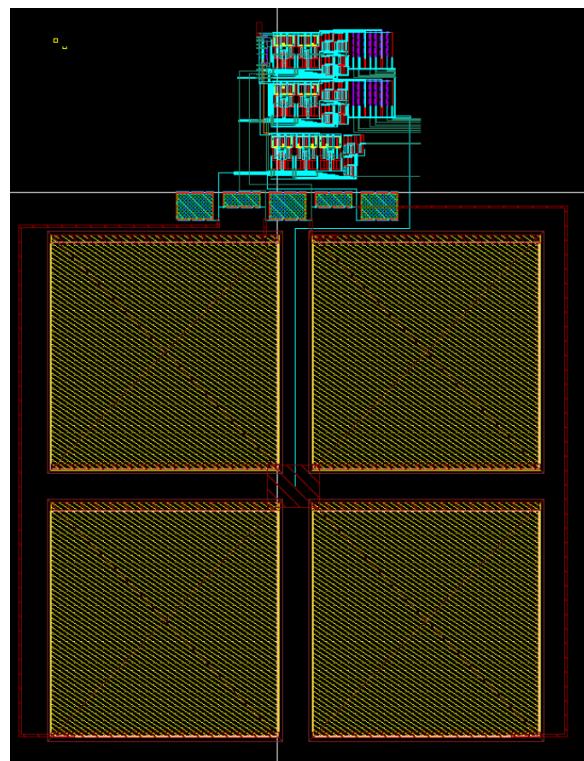
3. SubADC 版图



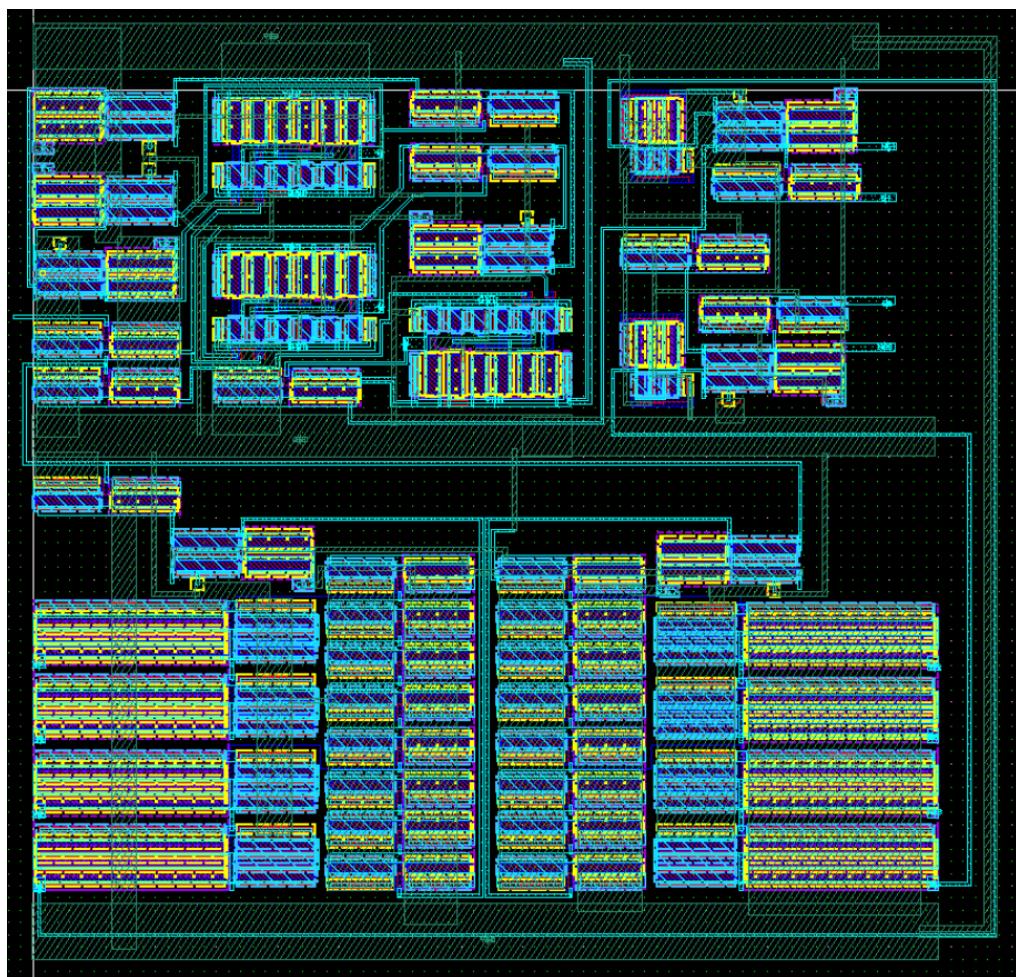
4. BackendADC 版图



5. 加上 MOS CAP 后 subADC, backendADC 版图



6. 时钟产生电路版图



7. 同步输出电路版图

