A Compiled 3.5fJ/conv.step 9b 20MS/s SAR ADC for Wireless Applications in 28nm FDSOI

Carsten Wulff*†,Trond Ytterdal*
*Norwegian University of Science and Technology, Email: carsten.wulff@ntnu.no
† Nordic Semiconductor

***Abstract*—A low power 9-bit compiled SAR ADC is presented. The ADC is compiled from a netlist, rule file, and object definition file into a DRC/LVS clean layout and schematic in 28nm FDSOI. The presented ADC also includes a comparator clock generation loop that use the bottom plate of the CDAC. The proposed compiled ADC achieves a FoM of 2.7fJ/conv.step at 2MS/s, and 3.5fJ/conv.step at 20MS/s with an area of 0.00312mm^2. To demonstrate process independence the same SAR architecture is compiled in 28nm FDSOI with IO transistors.**

I. Introduction

A low-power, medium resolution ADC is a key building block in Bluetooth® Low Energy receivers. Reducing design time of ADCs is an active research area, and recently an ADC was compiled from pure verilog code [1]. State-of-the-art analog layout generation, as covered in [2], however, is not widely adopted. This work presents a method where the physical layout is not generated based on drawn schematics, but rather the SAR ADC is described using an approach borrowed from object oriented programming. The presented method uses a limited set of circuit blocks, a low complexity SAR ADC architecture, and propose a technology independent method for generating a DRC/LVS clean schematic and layout in multiple technologies. To the best of the authors knowledge this work presents the most efficient compiled ADC and achieves a FoM comparable to the state-of-the-art [3] with a smaller area, making it perfectly suited for Bluetooth® Low Energy receivers. In this work the technology independence has been modeled with a tapeout of one SAR ADC in 28nm FDSOI with core transistors and another in 28nm FDSOI with IO transistors. The presented work also includes an asynchronous comparator clock generation loop that use the bottom plate of the CDAC.

II. Proposed Design

Two clocks are commonly used in SAR ADCs, one to set the sample rate, which is applied externally to the ADC, and one bit-cycling clock, which can be generated locally in the ADC. The comparator is usually reset during sampling, and it trigger when the sample clock turns off. It is common to use detection methods to determine when the comparator has made a decision, and use this to trigger a reset of the comparator. Such self-timed loops are common in prior art Fig. 1a). One of the challenges in a self-timed loop is to allow sufficient time for the capacitor DAC (CDAC) to settle. Each time the comparator makes a decision, one (or more) of the capacitors are switched and the resulting transient must settle to the accuracy of the ADC. It is possible to use circuit techniques (i.e. delay cells, or replica delays) to adjust the time between comparator decisions. Common to these techniques is that they try to model the required CDAC settling time, and allow enough time before the comparator makes the next decision. This work use the bottom plate of the CDAC directly in the comparator clock generation loop to improve robustness, and ensure sufficient time for CDAC settling.

III. SAR Architecture

The SAR ADC design is shown in Fig. 2, it has NMOS boot-strapped input switch, comparator with kick-back compensation [4], and metal-oxide-metal finger capacitor DAC.

The SAR logic stage in Fig. 3 consists of three parts; enable logic, CDAC state control, and clock generation loop. Enable logic consists of transistors M1 - M8, CDAC state control consists of transistors M9 - M16, and clock generation loop consists of transistors M17-M19 and the logic gates. The signals P and N are the positive and negative outputs from the comparator. CK is the sample clock. EI is the EO signal from the previous stage, where EO is the enable output to the next stage, and the first stage $EI = not(CK)$. The signals $CP0$, $CP1$, $CN0$ and $CN1$ are the control signals connected to the bottom plate of the capacitors in the CDAC. Signal CI is the bit-cycle clock input from previous stage, and for the first stage $CI = 0$. CO is the bit-cycle clock output to the next stage. The CO from stage 1 is connected to the CI of stage 2. The next sections explains in detail how the SAR logic works.

A. SAR logic reset

During sampling of the ADC input the SAR logic stage is reset ($CK = 1$). In the enable logic of the first stage $EI = not(CK) = 0$, and node $A = 1$, while $EO = 0$. Thus $EI = EO = 0$ of all subsequent stages. The CDAC state control has $CP0 = CN1 = 0$ while $CN0 = CP1 = 1$. In the clock generation loop of the first stage $CI = 0$, and since node $B = 0$, then $CO = 0$. Accordingly $CI = CO = 0$ for all subsequent stages.

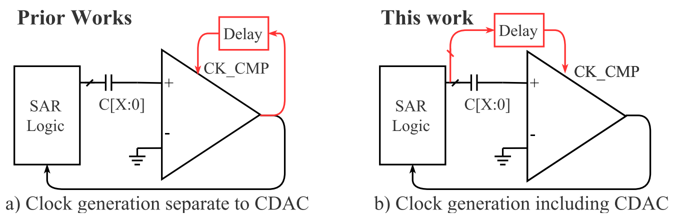

Fig. 1. Comparator clock generation in a) prior works, b) this work

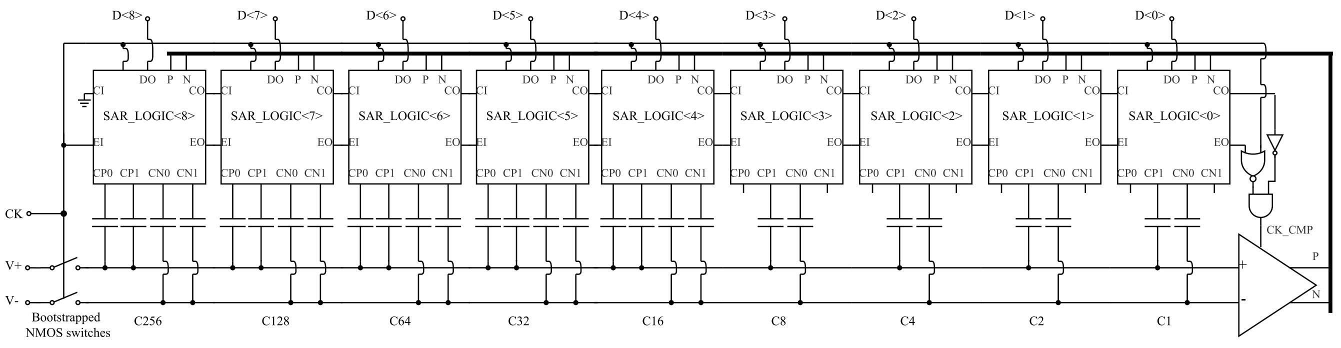

Fig. 2. Block diagram of proposed design

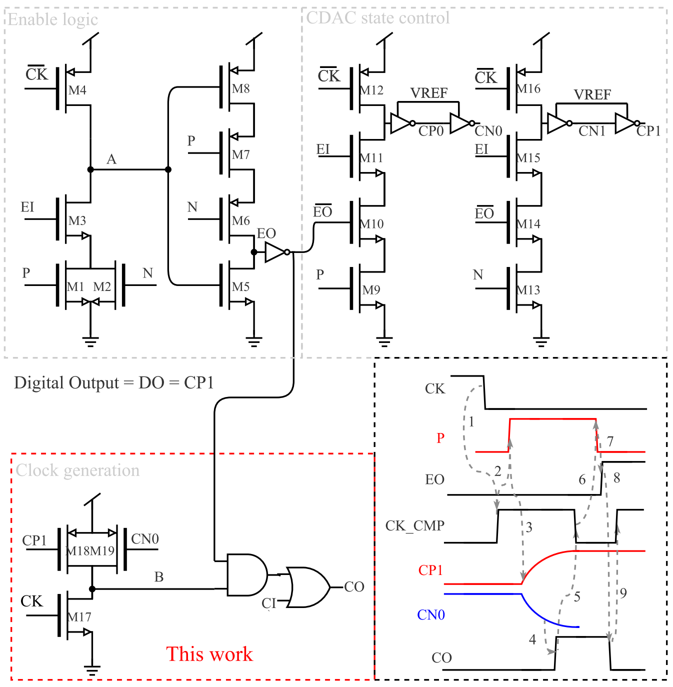

Fig. 3. SAR logic schematic

B. Comparator decision

The first comparator decision is initiated by $CK = 0$ (steps 1 and 2 in Fig. 3), at that point the latch (M1-M4) in the enable logic is armed, and as soon as $(P||N) = 1$, then $A = 0$. This arms the second latch (M5-M8) in the enable logic. Still $EO = 0$. The CDAC state control is also triggered by $(P||N) = 1$ (step 3 in Fig. 3), if $P = 1$ then $CP0 = 1$ and $CN0 = 0$. If $N = 1$ then $CN1 = 1$ and $CP1 = 0$. This either adds, or subtracts charge from the top plate of the CDAC. In this work symmetric switching (thus four CDAC control signals per bit) is used in the five MSBs to avoid a common mode shift, while the last four LSBs use monotonic switching, since the change in common mode is small. For MSB the transition of $CP1, CP0, CN0, CN1$ can be slow due to the high capacitance, especially if the SAR runs at a low voltage since the inverters will have lower driver strength. The advantage of this work is that the transition of $CP1$ and $CN0$ is included in the bit-cycling clock generation loop. Since either $CP1$ or $CN0$ is guaranteed to transition from high to low, these signals can be used to trigger comparator reset. When either $CP1$ or $CN0 = 0$ then M18 or M19 turns on, and sets node $B = 1$, and consequently $CO = 1$ (step 4 in Fig. 3). The delay from $P||N = 1$ to $CO = 1$ will depend on the process corner, CDAC capacitor variation, temperature, and supply voltage. As a result the CDAC will be given sufficient time to settle completely, independent of process, voltage, and temperature (PVT). The maximum sample rate will depend on PVT, thus in an industrial design one must ensure sufficient margin for sample rate over PVT. The SAR architecture in this work is well suited to a calibration loop where the supply voltage is adjusted until EO of the last logic stage is detected. In a Bluetooth® Low Energy system this can be implemented at radio power up. In this work, however, the voltage is fed from an external source, and the supply voltage calibration is a manual process. Another advantage of the clock generation loop in this work is that CK_CMP will speed up for LSBs, since $CP1$ and $CN0$ will transition faster for LSB stages, and the ripple path from CI to CO to comparator is shorter.

C. Comparator reset

The comparator in Fig. 2 is reset when $CK\_CMP = 0$, which occurs when $CO = 1$, since $CK = 0$, and for the last stage $EO = 0$ (steps 5 and 6 in Fig. 3). The comparator will set signals $P = N = 0$, which turns on transistors $M6$ and $M7$, and sets $EO = 1$ (step 7). This enables the next stage. It also locks the state of the CDAC state control, since M10 and M14 turn off. Also, $CO = 0$ and in the end $CK\_CMP = 1$ (step 8 and 9 in Fig. 3), which clocks the comparator once more, and the next bit is decided. The bit-cycling ends when $EO = 1$ for the last stage.

IV. Compiler

Schematic capture, simulation, physical layout, and parasitic netlist simulation, is a time consuming endeavor for ADCs. For charge based SAR ADCs the small capacitance's make them sensitive to poor physical layout. It is common to spend time tracking down fF parasitics, and redoing physical layout. The design cycle would be shorter if one could generate DRC/LVS clean layout in minutes, instead of days. Rapid ADC generation would allow for more iterations before tapeout, and could even allow automatic exploration of the design space. It could also enable rapid porting to a different technology. In this work the SAR architecture and logic is described in a standard SPICE netlist. An object definition file is used to describe how to route the physical layout of the SAR. A simplified technology rule file contains the necessary design rules to adapt the SAR to a different technology. A compiler, currently written in Perl, compiles the three input files into a SKILL file and a GDSII file. The SKILL file can be loaded in Cadence Virtuoso to generate the schematics and layout. The compiler generates DRC and LVS clean layout for both 28nm FDSOI IO transistors, and 28nm FDSOI core transistors. The next sections explain the input files to the compiler.

A. Rule file

The rule file contains technology specific information, like GDSII layer numbers, device names and port names in the PDK, what cut layers provide connection between routing layers, and design rules. For 28nm FDSOI there are 140 unique rules, but 95 of those are for the different metals and cuts (spacing, width, enclosure). All rules are described as a multiple of a technology $gamma$, where the $gamma = gatelength/6$. In addition the rule file defines a $grid$ size.

B. Netlist

A standard SPICE netlist defines the connectivity of all blocks in the SAR, as shown in Listing 1. The placement in the layout is determined by the order of instances in the SPICE file. The first element is placed in column zero and row zero, if the next block is of the same type the row number is incremented. If the next block is of a different type the row number is reset and column number is incremented.

Listing 1. Definition of inverter
```
.subckt IVX1_CV A Y AVDD AVSS
MN0 Y A AVSS AVSS NCHDL
MP0 Y A AVDD AVSS PCHDL
.ends IVX1_CV
```

C. Object definition file

The object definition file contains all instructions for routing the design, and information to generate transistors and capacitors. The transistor is described as pure ASCII as shown in Listing 2. The ASCII gives a NxM matrix where each cell is one grid in size. A $-$ character means empty space, x means the grid is filled. m means the grid is filled horizontally, but the height of the rectangle is the minimum routing width for that layer. For w the grid is also filled horizontally, but the height of the rectangle is the normal routing width for that layer. K, C, and c make cuts (vias) to the next layer. D, G, S, B are the familiar ports of the transistor. The object definition file uses an object oriented design approach, for example the PMOS and NMOS inherits the DMOS, and only need to add the necessary implant layers. This object oriented definition design approach allows complex circuits to be built from simple modification of sub blocks.

The inverter in Fig. 4 needs two routes, one for the PO gate, and one for the drain. The first routing command under IVX1_CV in Listing. 2 means “Route in metal 1, with net name Y. Find the D port on all instances called MN and route with a right, up or down, right path to port D on all instances called MP”. The complete object definition file that was used to generate the core transistor SAR ADC in this work consists of 1266 lines of code, but that file generates 21 different SAR variants, with 8-bit, 9-bit, 10-bit and 11-bit CDACs.

Listing 2. Defintion of a transistor in ASCII
```
{ "name" : "DMOS" ,
  "fillCoordinatesFromStrings" : [
      [   "OD",
          "------------------xxxx",
          "----xxK-----------xCxC",
          "----xxx-----------xxxx",
          "----xxK-----------xCxC",
          "------------------xxxx"
      ],
      [   "PO",
          "-mmmmmmmmmmmmm--------",
          "----------------------",
          "-mmmmmmmmmmcxc--------",
          "----------------------",
          "-mmmmmmmmmmmmm--------"
      ],
      [   "M1",
          "------------------xxxx",
          "----wDww----------xxxx",
          "-----------wGww---xBxx",
          "----wSww----------xxxx",
          "------------------xxxx"
      ]
  ]
},
{ "name" : "PCHDL",
  "inherit" : "DMOS",
  "beforePlace" :{
    "addEnclosures" : [["OD",["PP"]]]
  }
},
{ "name": "IVX1_CV" ,
  "beforeRoute" : {
    "addDirectedRoutes" : [
    ["M1","Y","MN:D-|--MP:D"],
    ["PO","A","MN:G-MP:G"]
    ]
  }
}
```

D. Porting to different technology

In this work we model a different technology by using core transistors in one SAR (28nm gate length), and IO transistors another SAR (180nm gate length). To port the SAR ADC from core transistors to IO transistors requires some changes to the compiler inputs. The metal spacing was changed in the rule file, the unit transistor (DMOS) was redefined, and PMOS and NMOS was modified to change the implant layers. There were no changes to the SPICE netlist. Less than 5% of the lines in the input files required changes to generate the IO transistor SAR from the core transistor SAR input files.

E. Compiler

A compiler, currently written in Perl, parses the rule file, netlist file, and object definition file and generates SKILL and GDSII. The Perl version of the compiler is closed source, but there is an ongoing initiative to make a open source version [5]. The compiler does not contain any auto-router features, but simple routing rules, similar to an instruction set.

V. Measurements

The prototype is fabricated in a 28nm FDSOI process with an area of 80μm x 39μm = 0.00312mm^2 for the core transistor SAR: including all SAR logic, comparator, CDAC, and input

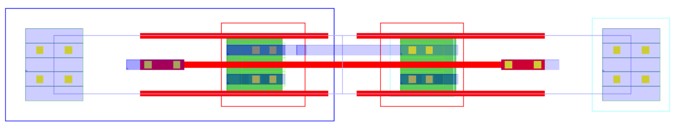

Fig. 4. Layout of an inverter.

switch, as shown in Fig. 6. The core transistor SAR ADC in this work achieves a FoM[1] of 3.5fJ/conv.step at 20MS/s, Fig. 5 (a,e), and 2.7fJ/conv.step at 2MS/s, Fig. 5(b), placing it among the most power efficient and area efficient ADCs for its sample rate and resolution. The spectra of the IO transistor SAR is shown in Fig. 5(c). The proposed 28nm FDSOI core transistor SAR ADC is functional from 80kHz - 80MS/s and operates from a single 0.4V-1.1V supply voltage as shown in Fig. 5(d). Table I shows a summary and comparison of key measurements with prior-art.

VI. Conclusion

A low power 9-bit compiled SAR ADC has been presented with a comparator clock generation loop that includes the bottom plate of the CDAC. The ADC was compiled from text input files into a DRC/LVS clean layout and schematic in 28nm FDSOI. The proposed ADC achieves a FoM of 3.5fJ/conv.step at 20MS/s with an area of 0.00312mm^2, and demonstrates that a compiled SAR ADCs can achieve state-of-the-art performance.

Acknowledgment

Partial funding for this work was obtained from the Norwegian PhD Network on Nanotechnology for Microsystems, which is sponsored by the Research Council of Norway, Division of Science, under contract no. 221860/F60

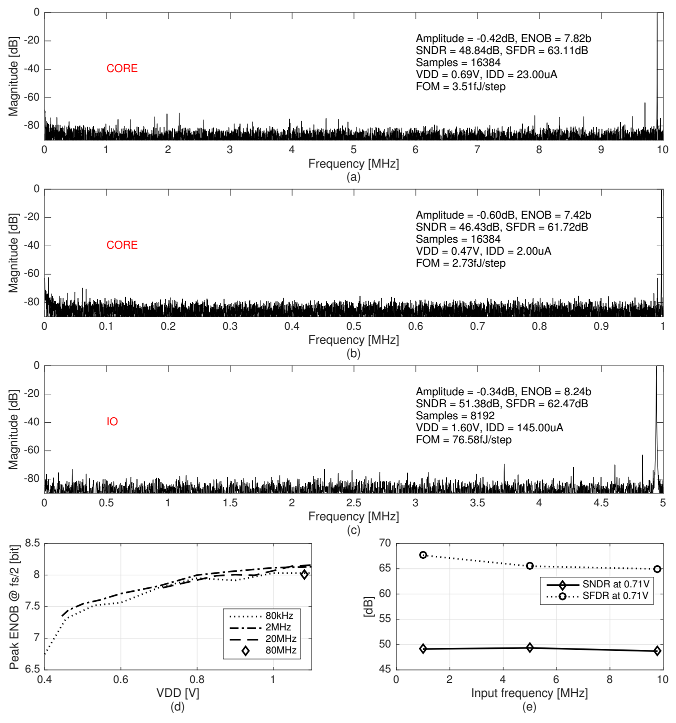

Fig. 5. Measurements of a) spectra of core SAR 20MS/s, b) spectra of core SAR 2MS/s, c) spectra of IO SAR at 10MS/s d) ENOB as a function of supply voltage for core SAR and e) SNDR and SFDR as a function of input frequency for core SAR at 20MS/s.

[1]$FOM = P/(2^{ENOB} * fs)$

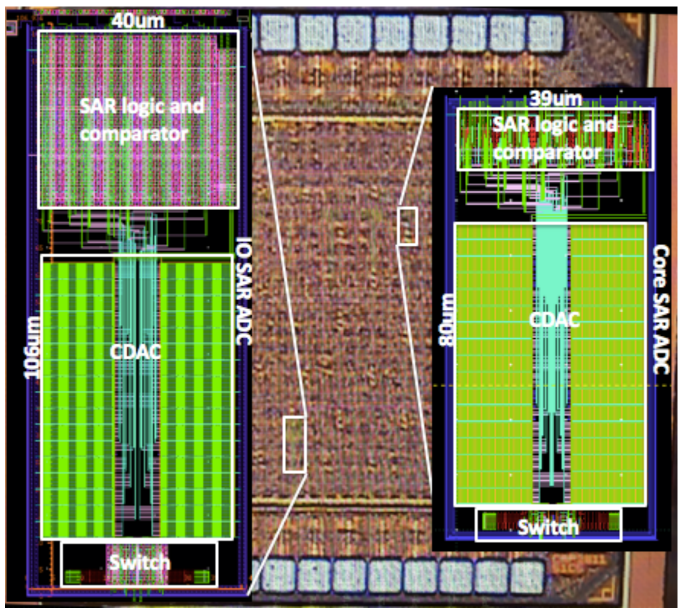

Fig. 6. Chip micro-graph with layout

TABLE I. Comparison to prior art

	[1]	[3]	This work		
Technology (nm)	90	90	28 FDSOI Core		28 FDSOI IO
Fsample (MS/s)	21	2	2	20	10
Core area (mm^2)	0.18	0.047	0.0032		0.0042
SNDR (dB)	34.61	57.79	46.43	48.84	51.38
SFDR (dB)	40.81	72.33	61.72	63.11	62.47
ENOB (bits)	5.45	6.7 - 9.4	7.42	7.82	8.24
Supply (V)	0.7	0.7	0.47	0.69	1.6
Ref. pwr (μW)	-	-	0.47	6.53	-
Other pwr (μW)	-	-	0.47	9.34	-
Total pwr (μW)	1110	1.64 -3.56	0.94	15.87	< 145 [2]
FoM (fJ/c.step)	838	2.8 - 6.6	2.74	3.51	< 76 [2]

References

[1] S. Weaver, B. Hershberg, and U. K. Moon, “Digitally Synthesized Stochastic Flash ADC Using Only Standard Digital Cells,” *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 61, no. 1, pp. 84–91, Jan 2014.

[2] R. Martins, N. Loureno, and N. Horta, “LAYGEN II - Automatic Layout Generation of Analog Integrated Circuits,” *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 32, no. 11, pp. 1641–1654, Nov 2013.

[3] P. Harpe, G. Dolmans, K. Philips, and H. de Groot, “A 0.7V 7-to-10bit 0-to-2MS/s flexible SAR ADC for ultra low-power wireless sensor nodes,” in *ESSCIRC (ESSCIRC), 2012 Proceedings of the*, Sept 2012, pp. 373–376.

[4] A. Varzaghani, A. Kasapi, D. N. Loizos, S.-H. Paik, S. Verma, S. Zogopoulos, and S. Sidiropoulos, “A 10.3-GS/s, 6-Bit Flash ADC for 10G Ethernet Applications,” *IEEE Journal of Solid-State Circuits*, vol. 48, no. 12, pp. 3038–3048, Dec 2013.

[5] C. Wulff. (2016) Custom IC creator. [Online]. Available: https://github.com/wulffern/ciccreator

[2]Due to limitations for the IO SAR it’s not possible to separate the current in pad drivers, and ADC current. The load on each pin is at least 4pF, as specified on the RTE1022 oscilloscope. Accordingly the current for the IO SAR is estimated at ADC current = 542μA - average(bit transitions x 10MHz x 4pF x 1.6V)